DOCUMENT RESUME

ED 086 179 IR 000 025

AUTHOR	Perone, Sam P.
TITLE	Introduction of Digital Computer Technology Into the Undergraduate Chemistry Laboratory. Final Technical Report.
INSTITUTION	Purdue Univ., Lafayette, Ind. Dept. of Chemistry.
SPONS AGENCY	National Science Foundation, Washington, D.C.
REPORT NO	NSF-TIE-GJ428-SPPCPU
PUB DATE	31 Aug 73
NOTE	157p.
EDRS PRICE	MF-$0.65 HC-$6.58
DESCRIPTORS	*Chemistry Instruction; Chemistry Teachers; *College Science; Computer Assisted Instruction; Computer Based Laboratories; Computer Programs; *Computer Science Education; Course Descriptions; Curriculum Development; *Digital Computers; Inservice Courses; Laboratory Techniques; On Line Systems; *Science Experiments; Undergraduate Study
IDENTIFIERS	PRTB; Purdue Real Time Basic System

ABSTRACT

The objective of this project has been the development of a successful approach for the incorporation of on-line computer technology into the undergraduate chemistry laboratory. This approach assumes no prior programing, electronics or instrumental analysis experience on the part of the student; it does not displace the chemistry content with computer related material. Readily implemented by an inexperienced undergraduate, it utilizes a real-time BASIC language and functionally-oriented general-purpose interface. The result is that the student can regard the laboratory computer as a very powerful routine experimental tool. An intensive three-week summer course was implemented for college instructors and practicing chemists to provide an introduction to digital instrumentation and the technology of digital computer implementation in the laboratory. Other developmental activities have included the design of the Purdue Real-Time BASIC software system and the writing of a number of experiments, both in digital logic and in chemical analysis using the computer, which will be incorporated into the proposed Laboratory Manual. The appendixes of this report include the texts of seven of these experiments. (SL)

ED 086179

FINAL TECHNICAL REPORT

Report Number - NSF-TIE-GJ428-SPPCPU

NSF Grant GJ-428 - "Introduction of Digital Computer Technology into the Undergraduate Chemistry Laboratory"

Principal Investigator: Sam P. Perone

Institution: Department of Chemistry
Purdue Univeristy
Lafayette, Indiana 47907

Grant Starting Date --- September 1, 1969

Grant Ending Date --- August 31, 1973

U.S. DEPARTMENT OF HEALTH,
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY.

IR000025

FILMED FROM BEST AVAILABLE COPY

Initial Objectives.

The objectives of the work conducted here were two-fold: (1) to develop a satisfactory and effective approach to the incorporation of digital computer technology into the undergraduate chemistry laboratory; and (2) to work in collaboration with Profs. C. E. Klopfenstein and C. L. Wilkins of the Universities of Oregon and Nebraska, respectively, to produce and publish a Lab Manual for computerized experimentation at the undergraduate level.

Major Accomplishments.

The most significant aspect of this project has been the development of a successful approach for the incorporation of computer technology into the undergraduate laboratory. This approach assumes no prior programming, electronics, or instrumental analysis experience. Moreover, a major prerequisite is that the chemistry content of the course work not be displaced by computer-related material. This has been accomplished by developing a laboratory computer system which is readily implemented by the inexperienced undergraduate. It utilizes a real-time BASIC language and a functionally-oriented general-purpose interface. The result is that the student can regard the laboratory computer as a very powerful routine experimental tool. He learns to use this tool properly and effectively, and yet the attention is focused on the chemistry or quantitative measurements to be made. Finally, it should be pointed out that the approach developed here is readily transferable to other institutions, regardless of specific alternate hardware configurations. Thus, the laboratory manual under development should find a wide and accessible audience.

During the past year we have completed, in collaboration with Professors C. E. Klopfenstein (University of Oregon) and C. L. Wilkins (University of Nebraska), an approximately 375 page laboratory manual for instruction of undergraduates in laboratory computer use. All material has been tested, using the manual, in a formal course taught at the University of Nebraska-Lincoln during the Spring semester 1973.

At Purdue, we have developed several of the experiments and appendices for the lab manual. In addition, we have proposed arbitrary standard data acquisition terminology for the Real-Time BASIC language. Finally, we continue working to eliminate "bugs" from old experiments and develop new experiments for inclusion in the Lab Manual. Final manuscript typing has been begun at Nebraska, and a publisher will be selected late this fall after we have obtained responses from all those interested. This will result in publication of the manual in 1974.

Papers Published, Submitted, or in Preparation.

A. Published.

S. P. Perone, "A Laboratory Course on Digital Computers in Chemical Instrumentation", J. Chem. Educ., 47, 105 (1970).

J. F. Eagleston and S. P. Perone, "The Introduction of Digital Computer Technology into the Undergraduate Chemistry Laboratory", J. Chem. Educ., 48, 317 (1971).

J. F. Eagleston and S. P. Perone, "On-Line Digital Computer Applications" in Gas Chromatography. An Experiment for the Undergraduate Analytical Laboratory", J. Chem. Educ., 48, 438 (1971).

D. O. Jones, M. D. Scamuffa, L. S. Portnoff, and S. P. Perone, "On-Line Digital Computer Applications to Kinetic Analysis. An Undergraduate Experiment", J. Chem. Educ., 49, 717 (1972).

S. P. Perone, F. E. Lytle, and J. F. Eagleston, "The Introduction of Digital Computer Technology into the Undergraduate Chemistry Laboratory", Proc. of Second Annual Conf. on Computers in the Undergraduate Curricula, Dartmouth College, June, 1971, p. 277.

S. P. Perone, "The Use of Computers in Undergraduate Laboratory Instruction", Proc. of Conf. on Computers in Chemical Education and Research, DeKalb, Illinois, July, 1971, p. 4-15.

S. P. Perone, "Training Chemists in Laboratory Computing", Computers in Chemical and Biochemical Research, Vol. 1, C. E. Klopfenstein and C. L. Wilkins, eds., Academic Press, NY, 1972, p. 1.

S. P. Perone and F. G. Pater, "Computer-Controlled Colorimetry. An Undergraduate Experiment, "J. Chem. Educ., 50, 428 (1973).

B. In Preparation.

D. O. Jones and S. P. Perone, "Synthetic Signal Sources for On-Line Computer Experiments", J. Chem. Educ., In preparation, 1973.

S. P. Perone and P. Gaarenstroom, "Real-Time Computer Processing for Predictive Coulometry. An Undergraduate Experiment", J. Chem. Educ., In preparation, 1973.

Supporting Material.

Included as an addendum to this report are copies of all relevant materials produced by this research program at Purdue. The unpublished materials included will be incorporated into the proposed Lab Manual also.

Reprinted from:
COMPUTERS IN CHEMICAL AND
BIOCHEMICAL RESEARCH, VOL. 1
© 1972
Academic Press, Inc., New York and London

Training Chemists in Laboratory Computing*

S. P. PERONE

Department of Chemistry, Purdue University, Lafayette, Indiana

I. A Postgraduate Laboratory Course on Digital Computers in Chemical Instrumentation 2
  A. On-Line Computerized Experimentation in the Laboratory 3
  B. An Educational Program in Computerized Instrumentation 7
  C. Detailed Course Description 11
  D. Hardware for On-Line Experimentation and Interface Design 14
II. Introduction of Digital Computer Technology into the Undergraduate Chemistry Laboratory 17
  A. Approach 18
  B. Purdue Real-Time Basic System 19
  C. Experimental 27
  D. Results and Discussion 32
  References 35

---

* Reprinted in part from Perone (1970) and Perone and Eagleston (1971), by permission of the copyright owner.

All © material in this document
"PERMISSION TO REPRODUCE THIS COPYRIGHTED MATERIAL HAS BEEN GRANTED BY
The © owner for
Sam P. Perone
TO ERIC AND ORGANIZATIONS OPERATING UNDER AGREEMENTS WITH THE NATIONAL INSTITUTE OF EDUCATION FURTHER REPRODUCTION OUTSIDE THE ERIC SYSTEM REQUIRES PERMISSION OF THE COPYRIGHT OWNER."

I. A Postgraduate Laboratory Course on Digital Computers in Chemical Instrumentation

The chemist today finds himself in the midst of a dramatic revolution in scientific instrumentation. Digital electronics and laboratory-scale digital computers appear destined soon to dominate the whole area of chemical experimentation. This revolution has come about so abruptly that the practicing scientist recognizes a very real technological gap in his background. He realizes that computerized instrumentation can provide benefits in routine and research work—but he does not know how to get started. To compound his frustration, he even finds it difficult to communicate with the experts and computer manufacturers. Not only is the *jargon* of digital computer technology quite foreign to the uninitiated, but the fundamental concepts of digital instrumentation are basically unfamiliar. The scientist is accustomed to thinking of experimental data and instrumentation from an *analog* viewpoint, and the introduction to the digital world is not without some difficulty. Not that the material is difficult—it is really quite simple for the scientist trained in logic—but mastery of the material necessitates the generation of drastically different instrumental and experimental concepts than the scientist is generally accustomed to. The principles, methodology, and jargon of digital instrumentation are so different from those of analog instrumentation that even the instrument-oriented scientist has difficulty mastering the area independently. Moreover, the scientist's ability to use the large data processing computer is of little help in understanding *on-line* applications of the digital computer.

Because scientists have recognized this technological gap, many have taken steps to overcome it. This has required herculean efforts on their part, generally, since very little detailed information is available. Most of what is available is provided by computer manufacturers, and may require thorough familiarization with a particular computer for any appreciation. The most fundamental problem, however, is that an adequate appreciation of this new technology cannot be acquired through reading or attendance at technical meetings. Moreover, it is not sufficient to have a computer available in the laboratory which the scientist can play with in his spare time. Nor is it sufficient to develop programming skills which are not oriented toward on-line experimental applications. The most efficient way to develop the technological skills necessary for the incorporation of the digital computer in laboratory experimentation is by total involvement in an intensive training program which includes substantial "hands-on" experience with digital instrumentation and the digital computer in an experimental on-line environment.

Thus, a need has existed for some time now to provide an educational program concerned with digital computer instrumentation for the chemistry laboratory. This type program is urgently sought by the practicing scientist—i.e., one who has completed his formal education. Moreover, this type program should be particularly valuable to our future scientists—the graduate and undergraduate students in our colleges and universities. Such a program has been instituted in the Chemistry Department at Purdue University under the direction of this author. It is the purpose of this article to describe the basic philosophy of this program as well as to provide the reader with many of the details of its operation. In addition, an evaluation of the results of this program will be presented.

A. ON-LINE COMPUTERIZED EXPERIMENTATION IN THE LABORATORY

To put this entire presentation in perspective, it is necessary to provide some background regarding on-line digital computer technology. A common characteristic of the laboratory-scale digital computer is a relatively small core memory, usually providing 4 K or 8 K of random access storage of digital data and programs. A memory of this size, however, can be more than adequate for laboratory control operations, and with careful programming, some fairly sophisticated data processing.

A small digital computer may have an *instruction set* which includes the order of 50–100 *machine language* instructions. Each of these instructions corresponds to a specific binary coding which, when decoded by the computer, results in the execution of a fairly simple arithmetic or logical step. Examples of some simple machine operations are (1) binary addition of a datum in some memory location to the contents of an arithmetic register (accumulator);

(2) transfer of the contents of an accumulator to a memory location (and vice versa); (3) rotation of the binary digits ("bits") of the accumulator contents to the left or right; (4) and the application of logical tests such as determining if the accumulator is zero, non-zero, odd, even, positive, negative, etc.

Obviously, the repertoire of machine instructions includes some rather elementary operations. However, by developing appropriate "programs" composed of many of these elementary operations, the most sophisticated mathematical computations can be carried out. Since the computer can execute instructions so rapidly—the order of 10^5–10^6 instructions per second—it can complete complex computations with fantastic speed. For example, a program to multiply two *n*-bit integer values might require a program including 50 or 60 statements, but may require only about 200 μsec for completion. Thus, some 5000 multiplications per second could be accomplished. With hardware arithmetic units, perhaps 100,000 multiplications per second could be completed.

Thus, the digital computer is a very simple-minded device, which must be told how to accomplish even the most fundamental computations, but which can accomplish these operations with blinding speed. Moreover, it is a tireless machine that will be content to calculate endlessly and consistently. It is also a very versatile device, since it is programmable and capable of accomplishing an infinite variety of computational, logical, or control operations. Finally, it is a device that can (in fact, *must*) communicate in a variety of ways with the outside world. It is this characteristic that defines the computer as a general-purpose experimental device.

1. *Off-Line Computers*

The computer configuration with which most scientists are familiar is the *off-line* system. This configuration is diagrammed in Fig. 1. To use the computer in this configuration, the scientist typically will write a data processing program in FORTRAN or some other higher level computer language, run the experiment(s), manually tabulate the data from a strip chart recorder or oscilloscope trace, transfer the tabulated data to punched cards, add the data cards to the deck of program cards, transport the combined card deck to the computer center for processing, and then wait until the program has been executed and the results printed.

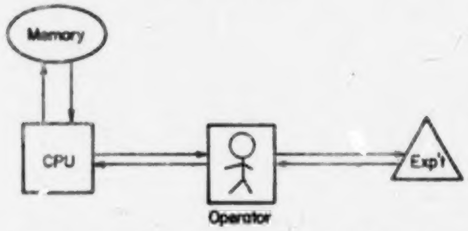


Fig. 1. Typical off-line computer configuration.

Turn-around times may vary from a few minutes to a few days, depending on the capacity of the computer facility and the number of users.

There are many variations of the above description of off-line computer usage. For example, the experimental system may include automatic data acquisition and digitizing devices which might store data on punched card, punched tape, or magnetic tape buffers which can then be transmitted more conveniently to the remote computer center. Alternatively, the laboratory may be equipped with a remote terminal (such as a Teletype) through which the investigator may enter his programs and data, as well as receive a printout of results from the computer.

The important common characteristic of all off-line computer systems, however, is that the experimental data are transmitted to the computer through some intermediate storage medium, and they are processed after some finite time delay has occurred. Depending on the modes of data acquisition and transmission to the computer, the turn-around time of the computer facility, and the speed with which the investigator can interpret the resulting printout, the time delay for experimental modifications based on the results of previous experiments can be excessive. Should this *reaction time* be a critical factor, an off-line computer facility may be inadequate for the particular experimental studies.

2. *On-Line Computers*

For the investigator who requires very rapid or instantaneous results from his computer system—for whatever reason—the solution may be to employ an on-line computer system. Figure 2 presents a block diagram for a typical on-line computer configuration. The most important distinction of this configuration is that there

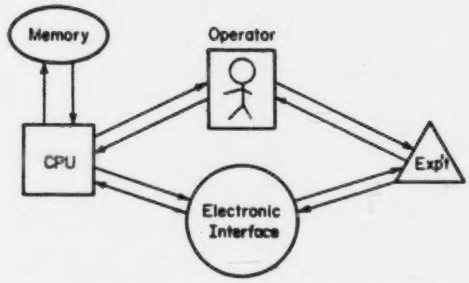


Fig. 2. Typical on-line computer configuration.

is a *direct* line of communication between the experiment and the computer. The line of communication is through an electronic interface. Data are acquired under computer control or supervision, and the program for data processing is either resident in memory or immediately readable into memory to provide for very rapid completion of the computational tasks. Results may be made available to the investigator quickly by means of teletype, line printer, oscilloscope, or other form of printout. In addition, the computer may be programmed to communicate directly with the experiment by controlling electronic or electromechanical devices—such as solid state switches, relays, stepping motors, servo motors—or any other devices which can be activated by voltage level changes.

The advantages of on-line computer operation can be summarized as the following: *Elimination of the middle man*, and the concomitant substitution of an electronic interface between the computer and the experimental system, which is much more compatible with the computer's characteristics and does not suffer from the inherent inadequacies of the human as a communication link. For example, the computer can accept input data at rates of the order of 10^5–10^6 per second, and can instantaneously transmit control information or commands back to the experimental system. If this line of communication were handled through the human investigator, as in the typical off-line configuration, the response time of the overall experimental and data processing system would be many orders of magnitude slower. Moreover, the electronic communication link is generally a more reliable, tireless, objective, and accurate information transfer medium.

The possibility of *direct computer control of the experiment* is a second advantage associated with on-line computer operation. This facility allows the complex logical and decision-making capabilities of the computer to be implemented for an infinite variety of laboratory automation or experimental design problems.

A third advantage of the on-line computer is that *real-time interaction between computer and experiment* is possible. That is, because the computer can make computations and decisions at speeds exceeding most ordinary data acquisition rates, it is possible for the computer to execute experimental control modifications before a given experiment has reached completion. These "real-time" operations allow the possibility of using the computer to monitor the progress of an experiment and modify that experiment to follow a pathway which provides more nearly optimum conditions and which could only have been arrived at as a result of some observation of the initial course of the experiment.

A *breakdown of the logistical barriers of the remote computer system* is an additional advantage of on-line computer operation. Because of the instantaneous, direct communication link between the experiment and the computer, the physical location of the computer with respect to the laboratory is relatively unimportant. Moreover, the mechanical and logistical roadblocks typically imposed by the computer facility toward the off-line introduction of data are irrelevant.

One very important aspect of using computers for on-line experimental work that should be emphasized here, however, is that some, if not all, of the programming must be done in machine (or assembly) language. That is, higher level languages, such as FORTRAN, do not have the capability for on-line operation with experimental systems. Moreover, higher level languages tend to be very inefficient in the use of memory space and in execution time. Both of these factors are critical when small computers are used for on-line work. Thus, the scientist wanting to use the small digital computer in the laboratory must become competent in programming in the fundamental machine language—a formidable task.

B. AN EDUCATIONAL PROGRAM IN COMPUTERIZED INSTRUMENTATION

The widespread use of digital computers on-line in chemistry laboratories will certainly revolutionize routine laboratory methodology. Today's scientist must learn to live with the digital computer and to use it profitably for laboratory work. On this premise we have developed at Purdue an intensive 3-week summer short course entitled "Digital Computers in Chemical Instrumentation." The course is designed for practicing scientists in colleges, universities, industrial laboratories, government laboratories, and hospitals, to provide an introduction to digital instrumentation and the technology of digital computer implementation in the laboratory. It is the details of this summer program which will be discussed here.

1. *Objectives*

The purpose of the 3-week intensive course developed at Purdue was to afford participants an opportunity to acquire a familiarity with digital instrumentation and digital computers in the laboratory. The course was designed such that students could develop, in a short period of time, the ability to write machine-language programs, to utilize the computer for typical laboratory input/output functions, and to design and implement software and hardware for interfacing real experimental systems to the computer for online operation. An essential feature of the course had to be the availability of several digital computers and many digital logic modules to provide a large amount of hands-on laboratory experience for each participant. To make the experience most worthwhile to the participants, the lecture and laboratory material were developed within the context of applications in chemistry and related sciences. (The course material was based on the assumption that participants had no previous experience with computers, digital logic, or electronics, but were generally familiar with chemical instrumentation and laboratory methodology.)

2. *Approach*

The basic philosophy of the 3-week summer course is simply to share with the participants the background, technique, and experience in computer instrumentation developed at Purdue and elsewhere by the faculty and graduate instructors of the course. For the participants to acquire the desired skills and background in the 3-week period, it is absolutely necessary to provide for as much individual access to computing equipment and digital instrumentation as possible and practicable. To provide this "hands-on" access to the equipment in the most efficient manner, a great deal of preparation outside the laboratory is required. Most importantly, a large ratio of teaching staff to students is provided, so that students can obtain help with a minimum of delay. At least one instructor for every four students in lab is the normal working ratio at all times.

The significance of the emphasis put on the efficient use of the available computing equipment cannot be overemphasized. The natural tendency of the novice programmer is to sit down at the computer console and try to compose a program. The students in our course learn very quickly that they must have their programs fully composed and punched on tape—with perhaps one or two alternates, in case the first does not work—before they even get to the computer console. When programs do not execute properly, they are encouraged to spend a minimum of formal lab time in time-consuming debugging operations. Thus, when a program is to be edited and reexecuted by the computer, a minimum of "hands-on" time will be consumed by the student trying to reason away the logical errors in his program.

Another important aspect of the course is that it is not limited to the concepts, principles, and techniques associated with the "small" digital computer. The formal lecture material discusses large, time-shared systems as well as small, dedicated systems. Moreover, one laboratory assignment involves writing a program and designing an interface to time-share several peripheral devices. The principles learned in this laboratory assignment are extended in lecture to describe the operation of larger time-shared computer systems.

3. *Requirements*

Equipment. An essential requirement for the presentation of the short course on computers in chemical instrumentation is the availability of sufficient hardware to allow adequate hands-on access by each participant. The minimum equipment provided for a twelve-man laboratory session includes: three digital computers with 8 K-word core memories, ASR/33 teletypes, and high-speed paper tape readers. One of these computers is equipped with a high-speed paper tape punch. In addition, each computer is completely equipped for high- or low-speed computer-controlled digital data acquisition with three 10-bit analog to digital converters (ADC) (each capable of data acquisition rates up to 20 or 30 KHz), and three frequency-selectable gated clocks for precise timing and synchronization of data acquisition and other experimental events. Two computers are equipped with high-precision (16-bit), high speed (20 μsec risetime) digital to analog converters (DAC). Two computers are equipped with oscilloscope display units interfaced to the computer through dual eight-bit DAC modules.

In addition to the computer equipment, each twelve-man laboratory has available six digital logic training stations. Each training station is equipped with versatile patchboard devices for assembling and testing digital logic circuits. Systems used for these stations include the Digital Logic Laboratory, Digital Equipment Corp., Maynard, Mass., and the Elite 3-a integrated circuit device tester, El Instruments, Inc., Derbyshire, Conn. Also with each station is a dual-trace wide bandpass oscilloscope. These stations are also used for assignments requiring interface design for computer-interfaced instrumentation.

Other items necessary to completely equip the laboratory include a variety of specially designed patchboard modules and associated electronics for instrumentation interface design, which are described in detail below. Also, at least one off-line ASR/33 teletype per every four students is provided for program preparation on punched paper tape.

Personnel. One of the most subtle, and yet most critical, requirements for presentation of the course described here is the availability of qualified individuals who understand the software and hardware aspects of computerized experimentation in the laboratory, and who can teach the material. At Purdue, we have been fortunate to find such personnel among our graduate students.

The explanation for the presence of qualified personnel here is twofold. We have several staff members who have become actively involved in on-line computer applications in their research, and we offer a formal one-semester course for graduate students designed to provide first-hand familiarity with on-line computer techniques in the chemistry laboratory. (This graduate level course, first offered informally in Spring 1968, formed the basis for the present intensive 3-week summer course.) Thus, by offering both the graduate course and the 3-week summer course, we have developed a unique group of well-trained teaching assistants.

C. DETAILED COURSE DESCRIPTION

1. *Schedule*

The summer course spans three weeks (or 15 working days). The first day is devoted to orientation and the first lab lecture. The next 13 weekdays are devoted to a rigorous schedule of laboratories, formal lectures, lab lectures, and open lab time. Each participant spends 1½ hours in lab lecture, 1½ hours at formal lecture, and 2½ hours in formal laboratory daily. In addition, each student has about two additional hours per day of "open" access to lab equipment so that he might work ahead or catch up. Labs also are open and supervised part-time on weekends. One day is devoted entirely to the demonstration of various on-line systems operating in the Chemistry Department at Purdue (including mass spectrometry, electrochemistry, flash photolysis, gas chromatography, and stopped-flow kinetic spectrophotometry). The last day is devoted to a "mini-symposium" on computer applications in chemistry, to which several expert outside speakers are invited.

2. *Content*

The formal lecture material covers the following three areas: digital logic and digital instrumentation; computer programming for on-line laboratory applications; and hardware and software design for instrumental interfacing, time-sharing, and computer control in the laboratory. A detailed lecture outline for the short course is given in Table I.

The formal laboratory assignments, likewise, were broken down into three categories: basic digital logic design and operation (4 days); basics of assembly language computer programming and operation (4 days); and interface design and computer programming for on-line applications (5 days). Lab lectures (1½ hours per day) were designed to prepare the student in advance for each laboratory assignment. Table II contains a detailed list of laboratory assignments.

TABLE I

SHORT COURSE LECTURE SCHEDULE

I. Introductory Remarks
II. Elements of Digital Logic
III. Digital Functional Units
IV. Number Systems; Binary Arithmetic, etc.
V. Boolean Algebra—Applications to Digital Circuitry
VI. Analog-to-Digital, Digital-to-Analog Conversion
VII. Computer Organization and Machine Language
VIII. Assembly Language Programming; Simple Arithmetic Routines
IX. Chemical Data Processing Routines
X. Programming the Computer for Input/Output with Standard Peripheral Devices
XI. Applications of Digital Logic in Chemical Instrumentation
XII. Digital Logic in Experimental Timing, Control, and Synchronization for Computer Interfacing
XIII. Data Acquisition and Real-Time Computer Control in Experimental Systems
XIV. Time-Sharing Systems; Combining Languages; Large Computer Systems

TABLE II

SHORT COURSE LABORATORY ASSIGNMENTS

A. Digital Logic
    I. Flip-flops and counters
    II. Gates and fundamental logic circuits
    III. Boolean equations and logic design
    IV. Digital control, timing, and synchronization
B. Computer Programming Fundamentals
    I. Elementary machine, and assembly, language programming
    II. Simple arithmetic programs
    III. Chemical data processing algorithms
    IV. Input/output programming
C. Interfacing and On-Line Computer Programming
    I. Data acquisition from a transient experiment
    II. Digital input/output
    III. Data acquisition and processing of gas chromatographic data
    IV. Ensemble averaging
    V. Time-shared service of several peripheral systems
    VI. Multiplexing and digital-to-analog operations

3. *Laboratory Procedures and Organization*

The organization of the laboratory and lab assignments is certainly most critical to the successful presentation of the short course.

With the available equipment providing the basic limitation on laboratory size, the labs were organized to make most efficient use of the hardware. This was accomplished by, first, dividing each lab into two groups: one-half start out working with the computer equipment; one-half start out working with digital logic equipment. Four days are required to complete the basic assignments in each group. Then, the two groups of students exchange equipment, and the basic set of assignments are repeated during the next four days. Two students are assigned to one computer per lab; one student is assigned to one logic station per lab.

While the basic sets of laboratory assignments are being carried out, the logic group attends lectures II–VI (Table I), while the computer group attends lectures VII–X. These basic lectures are primarily oriented towards the assigned laboratory material and are repeated when the two groups switch assignments.

After all participants have completed the basic sets of logic and computer programming assignments, five days of laboratory work follow which are oriented towards the development of programs and interfacing for on-line experimental applications. Students may continue to work in pairs on these assignments using any of the hardware available in the lab to execute each assignment. The apportionment of the computer time must be carefully supervised by the lab instructors during this period so that the equipment is used efficiently. One approach found effective here was to make available a fourth computer system which was used only for assembling and editing of program tapes. This freed the other computer systems for program execution and experimental design.

An interesting innovation was introduced during one recent course. Through the generosity of several computer manufacturers (Hewlett-Packard, Palo Alto, Calif.; Digital Equipment Corp., Maynard, Mass.; and Varian Data Machines, Palo Alto, Calif.) several different computer systems were made available during the last week of the course so that students might have the option of working with alternative equipment. The only requirement imposed was that they complete all normal course assignments up through at least the basic set of on-line experiments (C-I, C-II, Table II). Thus, confident that they had been exposed to all the basic technology we would hope to provide, we could turn students loose on unfamiliar equipment. Our objective was to prove to students that the operational differences between systems are not so great as they appear, once the fundamental principles have been mastered.

To make this possible, the additional equipment made available included two DEC PDP-8L computers with high-speed paper tape input/output, interfaced 12-bit ADC and four-channel multiplexer, and ASR/33 teletype; two DEC PDP-12 computers with full standard interfaced experimental input/output facilities (ADC, DAC, etc.), two LINC tape decks, oscilloscope display, and ASR/33 teletype; a Varian 620-i computer with provision for experimental I/O, oscilloscope display, and an ASR/33 teletype. In addition, two Hewlett-Packard computers (2116B and 2115A) with high-speed paper tape input and ASR/33 teletype were provided by the manufacturer for use along with the University-owned Hewlett-Packard systems.

The results of using the alternative computer equipment were very gratifying. Nearly all of the students who switched to the unfamiliar equipment were able to quickly master the new assembly language software and operating procedures. In fact, many were able to develop programming and interfacing for on-line operation. All this was accomplished with only a total of three days exposure to the equipment! These results seemed to prove that the necessity for exposure to one type of computer equipment in the laboratory part of the course is certainly no hindrance to the later use of other systems.

D. HARDWARE FOR ON-LINE EXPERIMENTATION AND INTERFACE DESIGN

Because it was unrealistic to assume that all students would be skilled in electronics, the development of lecture and laboratory material on interfacing experimental systems for on-line operation was particularly challenging. However, by building on the fundamentals of digital logic learned in the early part of the course, it is possible to adequately cover the topic of interface design.

The basic approach used involved recognizing that the digital computer communicates with the outside world by the execution of instructions which either cause binary voltage level changes to occur at specific external terminals, or which recognize voltage level changes effected by external devices. This understanding, coupled with the fact that the digital computer can neither accept nor output normal analog (continuously variable) signals, determine the type of interface functions required.

One primary group of interface elements to be considered are *translational* modules, such as analog-to-digital converters and digital-to-analog converters. In addition, such devices as level converters and analog amplifiers contribute to translational functions. Also important for consideration are multiplexers and signal transmission problems.

Those elements essential to proper communications are the logic, timing, and control components. These include flip-flops, gates, one-shot delays, Schmitt triggers, clocks, counters, analog switches, and electromechanical devices like relays and stepping motors. The student is made aware of the functional characteristics of all of these interface elements. (Many are actually used and tested individually in the laboratory.) He is then responsible for designing interfaces for assigned experiments by selection and combination of appropriate elements to provide both the translational and logical functions required. This means that he must be able to provide for the coordination and synchronization of the computer's program execution with the experiment and with the data acquisition system.

To implement these principles in the laboratory, we provide students with the building blocks with which the actual interface can be constructed in patchboard fashion, once the logical design is complete. Included in the patchboard interfacing package are a 10-bit (30 KHz) externally controlled ADC; a gated 10 MHz crystal clock with scaled outputs available in decade steps from 1 MHz to 0.01 Hz; a four-channel multiplexer; two sample/hold amplifiers; and 8 bits of binary control and logic I/O. Another type of patchboard interface panel is available which provides for 16-bit digital I/O only. The digital logic stations are available to allow incorporation of logic and timing elements (gating, time-delays, etc.) not included in the basic interface panel. All interconnections can be made externally with plug-in leads. (See Section II for a detailed view of interfacing hardware.)

1. *On-Line Experiments*

The ultimate objective of the laboratory work is for students to be able to carry out experiments on-line with the digital computer. Thus, the final set of laboratory assignments requires the student to design and implement the programming and interfacing for a variety of on-line experimental systems. Only the detailed characteristics and data acquisition requirements for each experiment are specified for each assignment.

Basically, two types of experimental systems were provided for laboratory assignments: synthetic and real. The synthetic systems were electronically simulated experimental outputs where the control, synchronization, and signal-handling problems were made nearly identical to several real systems. These electronic simulators were constructed on printed-circuit cards that could be plugged into the patchboard logic devices. Thus, patchboard interfacing could be implemented easily.

Simulated experimental output included an externally triggered rapid exponential transient decay voltage (time constant of 0.1 sec); an externally triggered repetitive voltage waveform with superimposed large amplitude random noise for signal-averaging studies; simultaneously generated multiple voltage waveforms for multiplexing experiments; and a synthetic typical gas chromatographic output including several peaks of varying shape and size.

The above-simulated systems provided convenient, reliable, and realistic on-line experimental experience for each student. The most obvious advantage was the freedom from chemical and instrumental uncertainties invariably associated with real chemical laboratory systems. On the other hand, it was desired to provide some experience with real laboratory systems since this appears to make the most lasting impression on students. Thus, laboratory gas chromatographs and a variety of sample mixtures were made available for students to apply their on-line data acquisition and processing skills. Appropriate amplification of the normal GC outputs could be provided by the selection of an operational amplifier-based plug-in module. The rest of the interface design was left to the student.

The minimum of assignments expected to be completed by all students included data acquisition from a simulated transient experiment, using computer-controlled constant or variable data acquisition rates, and computer-controlled data acquisition from an external digital counter. Beyond these minimal assignments students could carry out ensemble averaging, multiplexing, and gas chromatographic experiments. In addition, assignments involving computer-controlled digital-to-analog waveform generation and time-sharing of multiple external systems could be executed.

Many of the above described assignments required that *real-time* data analysis be incorporated into the programming. That is, some assignments required data analysis during data acquisition. Mastery of this type of programming is indicative of an accomplished experimental programmer.

2. *Results*

The primary result of the development and presentation of the course described here has been to make available to practicing scientists, regardless of background or previous experience, a rapid, effective introduction to computer technology in the laboratory. In addition, the instrumentation and educational techniques developed for the short course have been incorporated into a graduate-level chemical instrumentation course offered at Purdue.

Moreover, many of the university and college professors taking the summer course have been encouraged to develop similar computer-oriented course work at their respective institutions. Thus, the development of curricula which include the on-line implementation of computers in undergraduate and graduate scientific work has been fostered in a most efficient manner.

II. Introduction of Digital Computer Technology into the Undergraduate Chemistry Laboratory

It should be clear that the chemist of the near future will be required to understand and properly use digital instrumentation and digital computers in the laboratory. Furthermore, educators at colleges and universities will need to develop undergraduate curricula providing the required background in computerized chemical instrumentation. This section describes the first steps in a long-range program involving the introduction of on-line computer applications in the undergraduate chemistry laboratory at Purdue. The introduction has been made as part of an introductory analytical chemistry course dealing with quantitative chemical measurements (Junior level at Purdue). A primary objective was that none of the chemical and instrumental principles normally developed in the course be eliminated, but rather that these be augmented by the incorporation of the on-line computer into many of the laboratory experiments. The manner in which such a program may be made feasible is described below.

A. APPROACH

It was our intention to introduce computer technology into the undergraduate chemistry laboratory without slighting the chemistry content. The approach has been to continue to include the same basic set of laboratory experiments—except that many are now designed to require on-line communication with the digital computer. The student is required to understand and execute the laboratory experiment as usual—but, in addition, he is required to design the appropriate programming and communication elements for optimum interaction between the computer, experiment, and experimentalist. Thus, we envision the student's achievement to be one of *experimental design,* and the *proper utilization* of computing equipment in the solution of measurement problems in chemistry. Specific aspects emphasized include the general-purpose nature of the digital computer (i.e., how it can be applied to a large variety of experimental problems with modifications in programming); the computer as a control element in experimentation; the rapid response of the computer, allowing real-time interaction with experiments—with reaction times several orders of magnitude faster than the human operator; and the computer as an integral part of the chemical instrumentation.

To accomplish these objectives, it was necessary to recognize, first of all, what could *not* be expected within the scope of this course. (1) We could not educate our students in the intricacies of machine-language programming, and (2) we could not provide them with the electronics background requisite for the sophisticated design and construction of interfacing between computer and experiment.

What *could* be done is to take advantage of students' previous exposure to off-line computer programming (with FORTRAN usually), which many, but not all, students have obtained before the Junior year at Purdue. We could also take advantage of their exposure to the fundamentals of electrical measurements in physics courses to instill in them the most elementary concepts of amplification, response, and noise. We could expect to provide them with the essential principles of digital logic required for interface *design*.

The most important requirement for the success of the program, however, was the development of two items: (1) a modified high-level programming language, which includes versatile data acquisition and experimental control subroutines with conversational mode calling sequences; and (2) a general-purpose hardware package for interface design. This apparatus would provide for patchboard incorporation of those digital and analog modules required to complete the interface between computer and experiment. The general-purpose interface package requires only that the student be able to lay out the basic logic design, timing sequence, and the analog amplification or attenuation required. He can then select appropriate pre-packaged plug-in units to implement his design. He does not have to tend to such details as level conversion, logic conversion, noise rejection, and other subtle design parameters. Yet, he will accomplish the most important part of the interface development—logic design.

B. PURDUE REAL-TIME BASIC SYSTEM

The high-level programming language chosen for development in this work was the BASIC language (Kemeny and Kurtz, 1967). There were several reasons for selecting this language: (1) It is *easy to learn*, generally requiring on the order of half a day exposure to develop a good working knowledge. (2) It is an algebraically oriented *conversational* language. (3) It is *interactive*. That is, the compiler is *interpretive*, and therefore compiles and executes programs line-by-line. This allows programs to be entered, executed, and edited on-line through a teletype terminal. It also provides for immediate turn-around and rapid error diagnostics. (4) BASIC is rapidly becoming a *universally acceptable* language. (5) BASIC is currently readily *available* at commercial time-share terminals. This allows the convenient and economical learning of the language by large numbers of students for later on-line experimental applications.

It should be noted that other languages (e.g., FOCAL[1]) are available which have comparably desirable characteristics. Even languages such as FORTRAN or ALGOL which are generally considered strictly for off-line applications may be appropriate, particularly for systems with "load-and-go" capability.

The software system developed at Purdue will be referred to as "Purdue Real-Time BASIC" (PRTB). The software includes, fundamentally, the BASIC compiler available from Hewlett-Packard Co., which manufactured the computers used in the undergraduate laboratory program (see Section II,C). The modifications generated here at Purdue have involved the development of a series of machine-language subroutines which are directly callable from the BASIC software, and which are designed to communicate in a variety of ways with experimental systems. The PRTB data acquisition and control software are described below.

1. *General Description of PRTB System*

Figure 3 illustrates the general nature of an on-line computer system. There is the laboratory experiment—the data from which the student desires to present to the computer for data processing—and the digital computer itself, which is capable of high-speed

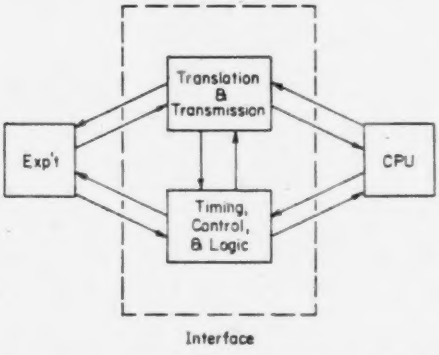


Fig. 3. Schematic diagram of on-line computer configuration.

---

[1] Trade-mark, Digital Equipment Corp., Maynard, Mass.

programmed computational and control functions. To operate the communication link between the computer and the experimental system, an electronic interface must be established. This interface accomplishes the functions of translation (analog-to-digital, digital-to-analog, decoding, logic conversions, etc.), timing, synchronization, and logical control. The programming language must take into account the nature of the electronic interfacing, and the interfacing must be designed with the fundamental characteristics and capabilities of the programming language in mind. The following section describes the subroutines developed for on-line experimentation within the PRTB system. A summary of these subroutines and characteristics is given in Table III.

2. *Experimental Input/Output Subroutines for PRTB*

Data Acquisition. Two different types of data acquisition subroutines are available within PRTB. They both work in conjunction with an external clock-controlled analog-to-digital converter. One is a subroutine (SB3) which should be called by the user's program whenever the computer should be waiting for the next data point to become available. When the next datum is digitized, SB3 takes the data point from the data acquisition device (ADC), converts the datum to floating point format, and stores it in the appropriate memory location for subsequent reference by the BASIC program. When SB3 is exited, the computer returns to the next statement in the BASIC program for execution. The new datum can be operated on; but, before the next data point can be acquired, SB3 must be called again.

A second type of data acquisition subroutine (SB7) is one that allows for the acquisition of a complete block of data before exit. The external clock is started and synchronized with data acquisition within the subroutine. The basic difference between SB7 and SB3 is that SB3 is called to acquire one data point at a time, and therefore allows for program statements to be executed during the time between acquired data points. Thus, an experimenter could devise a program that could process experimental data while the experiment was in progress and data were being acquired. This is referred to as "real-time" data processing. However, because the computer program must involve the relatively inefficient execution of BASIC statements between data points, there is a more severe limit on the speed with which data can be acquired without the computer

TABLE III

DATA ACQUISITION AND CONTROL SUBROUTINES FOR PRTB

Subroutine	Function	CALL Format
SB1	Initialize I/O. Set up data storage address and acquisition rate for other subroutines. "X(*I*)" data variable. "*F*" = data acquisition frequency.	CALL (1,X(I),F)
SB2	Used to start the CLOCK. The output control bit (ENCODE) can be connected to some external logic preceding the CLOCK ENABLE input. (NOTE: SB2 must be preceded by SB1.)	CALL (2)
SB3	Waits for ADC FLAG, indicating conversion completed. One datum is then taken from the ADC, converted to floating point, and saved as X(*I*).	CALL (3)
SB4	When called this subroutine waits for switch 0 of the console switch register to be set to a "1" before continuing with the next statement in the BASIC program.	CALL (4)
SB5	Outputs one control bit (ENCODE) and then waits for an event FLAG on the specified I/O channel, "*C*," before continuing. FLAG and ENCODE are cleared before exit.	CALL (5,C)
SB6	Waits for FLAGS on the ADC channel. When FLAG is set, the next statement in the BASIC program is executed. (Must be preceded by SB1 and SB2 if internal CLOCK is used to generate FLAGS.)	CALL (6)
SB7	Takes in complete block of data before returning to BASIC program. (Max. of 250 points. Must be preceded by SB1 each time it is called.) "*T*" sets total No. of data points taken. Synchronized with start of experiment through external gating of the ENCODE output. Up to 20 KHz data acquisition rate possible.	CALL (7,T)
SB8	Causes output bit No. "*Z*" to be set TRUE on specified I/O channel, "*C*." *Z*, *C* specified in decimal. All other bits will be cleared.	CALL (8,Z,C)
SB9	Causes output of analog voltage through the DAC. "*D*" = mV output.	CALL (9,D)

TABLE III (*Continued*)

Subroutine	Function	CALL Format
SB10	Clears ADC channel. External Flip-flop driven by ENCODE will be cleared also. This can be used to turn off clock, stop experiment, etc.	CALL (10)
SB11	Causes output of binary voltage pattern, "*Z*," on specified output channel, "*C*." Binary pattern and channel specified in decimal.	CALL (11,Z,C)
SB12	Inputs status of 16-bit register on channel "*C*." The floating point equivalent is saved in memory as "*Z*."	CALL (12,Z,C)
SB13	Inputs status of a specific bit (No. "*Z*") on channel "*C*." Value of "*S*" is made "1" or "0" accordingly.	CALL (13,Z,S,C)

getting out of synchronization with the experimental timing. In fact, SB3 is designed to detect when the computer got out of synchronization; an error message will be typed, and the computer will halt. The user will then have to revise his program to require less real-time processing. With SB7, on the other hand, a complete block of data points is acquired before the subroutine is exited and control returned to BASIC. Therefore, the timing is limited by the efficiency of the machine-language programming developed within SB7. For our computer systems and software, it is possible to acquire data at rates as great as 20 KHz with SB7, but no real-time data processing is possible.

The limiting data rate when using SB3 is about 500 Hz, with a minimum of real-time data handling. If several BASIC computations are to be executed between data points when using SB3, the limiting data acquisition rate may be the order of 1 to 50 Hz. However, this is generally more than adequate for most experiments in the chemistry laboratory.

Experimental Control and Logic. In addition to data acquisition, there are several other ways in which the computer can communicate with the experimental system. One of these is through a digital-to-analog converter (DAC), where the digital output of the computer is converted by the DAC to an equivalent analog voltage level. One of the PRTB subroutines (SB9) provides the capability for driving the DAC, the output of which can be connected to external experiments. It is possible to generate a continuous voltage waveform output from the DAC by mathematical generation within BASIC of the discrete points making up the waveform and transmitting these through SB9 to the DAC in a repetitive fashion synchronized with an external clock.

Subroutines SB8 and SB11 allow the programmer to utilize specific output bits on a selected I/O channel to control external devices. There are 16 output bits available for this function. With SB8, the user can select which bit he wants to set by specification of the bit number. The setting of one of these output bits causes a corresponding binary voltage level change at the specified output terminal, and this can be used to close or open switches, start or stop experimental events, light indicator lamps, etc. SB11 allows the programmer to output a 16-bit binary voltage *pattern* with any simultaneous combination of "1"s and "0"s he chooses. This binary pattern is selected by including in the subroutine call the decimal equivalent of the binary number to be generated. Thus, more than one event can be controlled simultaneously.

SB12 and SB13 provide digital *input* information for program "sensing" of external situations. Thus, 16 binary voltage input terminals are available to the user, the status of which are acquired by either subroutine. SB12 transmits to the BASIC program the numerical equivalent of the input 16-bit binary voltage pattern. Because the status of these bits can be set by external events, the computer could use this information to make appropriate changes in the data processing or control programming.

SB13 is similar to SB12, except that only the status of a single specified input bit is determined. This subroutine is useful to the student because he need only specify a bit number in the call statement to check the status of a bit.

Timing and Synchronization. The most fundamental operation which must be accomplished by the PRTB system is the generation of a *time base* for all experimental functions. This is accomplished by the incorporation of a fixed frequency crystal clock (10 MHz) into the interface hardware. Also included is electronic countdown logic to scale the output clock pulses down to a usable frequency range for chemical experimentation (100 KHz to 0.01 Hz). The countdown logic can be modified under program control to select any frequency within this which can be generated by a decade-and/or a 1,2,5-countdown sequence. The programmed clock output pulse train is then available to control the timing on the ADC, DAC, or any other external hardware. Also available simultaneously are synchronous clock pulses representing frequencies at the various stages of countdown. (These details are described in Section II,C.)

The clock is controlled by the PRTB software through subroutines SB1, SB2, and SB10. SB1 is the *initialization* subroutine. Through it the programmer specifies the clock frequency, within the limits outlined above. (SB1 is also used to specify the symbol assigned to the variable which will take on the values of the digital data acquired in SB3 or SB7.)

SB2 is called when the computer program decides that it is time to *enable* the clock. That is, an ENCODE bit is set to a "1" state on the data acquisition channel. This bit can be connected externally with a patchcord to the ENABLE terminal on the clock module (see Fig. 4a). Alternatively, the ENCODE bit may be brought to some external logic, the output of which will set the ENABLE clock input when other external events have occurred—like the start of the experiment. One possible external logic configuration is shown in Fig. 4b, where the ENCODE bit conditions one input of an AND gate. The other gate input is conditioned "TRUE" when the experiment has been initiated. The AND gate output will then enable the clock. At that point the countdown logic will be enabled and programmed clock pulses will begin to appear at the available outputs.

SB10 is called to disable the clock when the time base is no longer required. It simply "clears" the ENCODE bit on the data acquisition channel which has been used externally to enable the clock.

Another type of communication required between the experiment and computer is associated with *synchronization* between computer operations and external events. For example, if the computer has completed all preliminary program execution required before being able to accept experimental data, the user may choose to receive an output from the computer which may not only tell the experiment to start, but may also initiate external events associated with the experimental system. One subroutine available for this kind of operation is SB5. SB5 causes the output of an ENCODE bit on a selected I/O channel to change state (go to the "TRUE" level)

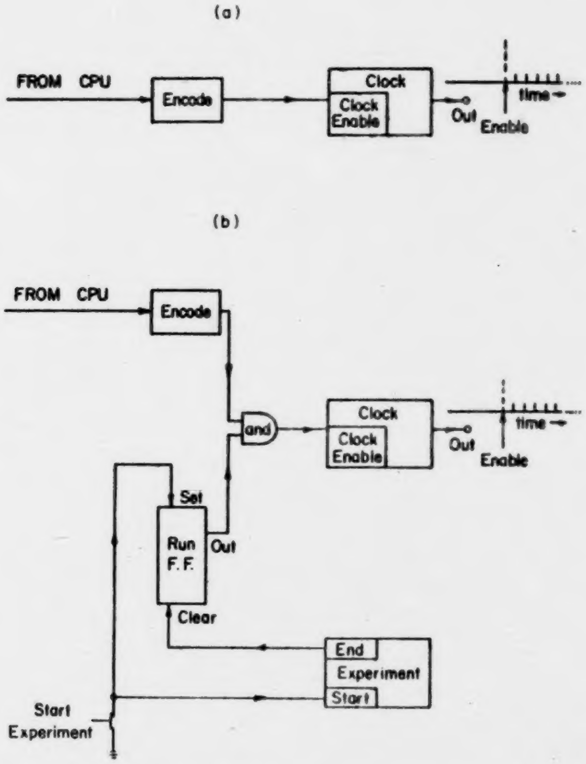

(a)
FROM CPU
Encode
Clock
Clock Enable
Out
time →
Enable
(b)
FROM CPU
Encode
and
Clock
Clock Enable
Out
time →
Enable
Set
Run F.F.
Out
Clear
End
Experiment
Start
Start Experiment

Fig. 4. Synchronization of clock output with computer and experimental events. (a) Direct computer control of clock enable input. (b) Clock enable controlled by AND gate output. Clock enabled when both computer command and experimental start are seen at AND gate inputs. The "RUN" flip-flop output follows the status of the experiment; it is set "true" when the experiment starts and "false" when the experiment ends. Thus, the clock is disabled when the experiment ends.

when SB5 is called. Then the computer waits within SB5 for an external event to occur which will cause a FLAG bit on the same I/O channel to change to a TRUE state. The subroutine detects this event and then allows the next sequential program statement to be executed. This next statement might conceivably be a call to SB2 which initiates data acquisition.

A simpler, but less precise, means of communication with the computer for the purpose of synchronization is provided by SB4. When this subroutine is entered, the computer simply waits until the operator flips a toggle switch corresponding to bit 0 on the computer console switch register. The computer then exits SB4 and executes the next program statement, which might be to call SB2 and start data acquisition.

Finally, a subroutine (SB6) is available which allows general-purpose timing functions. This routine simply waits for the FLAG bit to be set on the data acquisition channel, clears the FLAG, exits the subroutine, and the next sequential program statement is executed. The purpose of this subroutine is synchronization of program segments with the external time base. (Note that SB5 can be used, also, for general-purpose timing functions if the data acquisition channel is needed for other purposes. The user must simply connect the clock output to the FLAG input of the alternative I/O channel.)

C. EXPERIMENTAL

1. *Computer Instrumentation*

The digital computer system used in this work was a Hewlett-Packard 2115A, equipped with 8 K core memory, high speed paper tape input, ASR/33 Teletype, a 16-bit 20 μsec DAC, and an interfaced Tektronix Model 601 oscilloscopic display. In addition, the computer has an interfaced data acquisition system and a general-purpose experimental interface capability. (These are described in detail below.) The complete computer system is mounted in a cabinet which has roll-around capability. The software used for laboratory on-line operation (PRTB) utilized the Hewlett-Packard BASIC compiler, Hewlett-Packard No. 201112A. This software is made available from the computer manufacturer. The program listings of the *additions* to the Hewlett-Packard BASIC compiler made here to implement PRTB are available from the author upon request.

2. *Data Acquisition and General-Purpose Interface Hardware*

A schematic diagram of the data acquisition system operated in conjunction with PRTB is shown in Fig. 5. The analog-to-digital

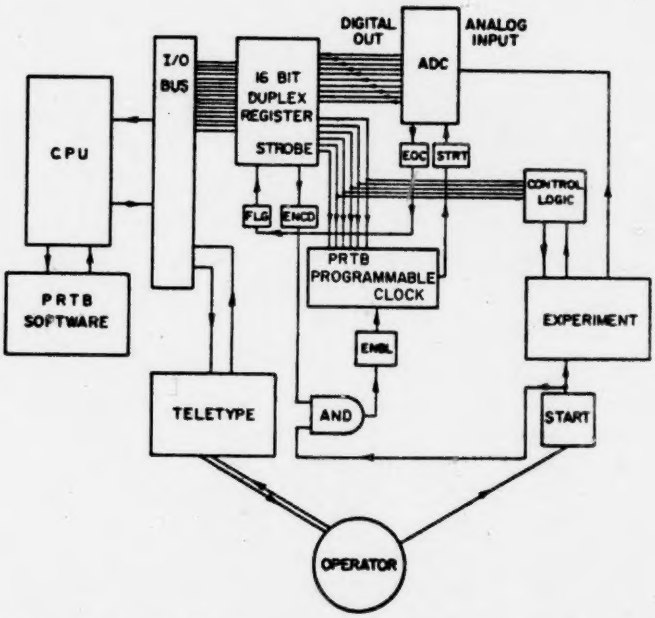
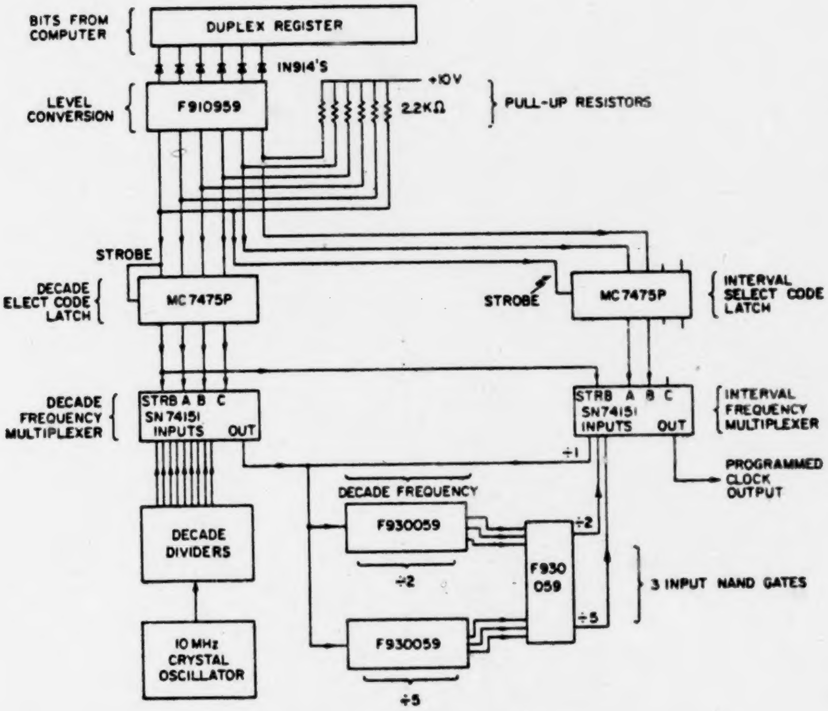

DIGITAL OUT
ADC
ANALOG INPUT
I/O BUS
16 BIT DUPLEX REGISTER
STROBE
CPU
EOC
STRT
CONTROL LOGIC
FLG
ENCD
PRTB PROGRAMMABLE CLOCK
PRTB SOFTWARE
EXPERIMENT
ENBL
TELETYPE
AND
START
OPERATOR

Fig. 5. Typical PRTB experimental data acquisition setup. Previously undefined symbols include: I/O Bus, Input/Output hardware of the computer, one channel of which is used for data acquisition system; 16-bit duplex register, a Hewlett-Packard #02116-6195 16-bit duplex interface buffer register card, also providing output ENCODE command and input FLAG signal.

converter used in the data acquisition system was manufactured by Digital Equipment Corporation (DEC), Maynard, Massachusetts (No. C-002, 10-bit 33 μsec conversion time, 0 to −10.23 V input range).

The programmable clock is constructed from a 10 MHz crystal-controlled oscillator scaled down to usable frequency ranges with medium-scale integrated circuit (MSI) programmable countdown logic. A schematic diagram of this module is given in Fig. 6, with a complete list of hardware components. Programmed clock pulses are available from 100 KHz to 0.002 Hz. Each decade frequency can be divided by 2 or 5. The specific frequency is selected by a 5-bit output word which is decoded in the clock module by two Texas Instrument SN74151 digital multiplexers. In addition to the pro-


BITS FROM COMPUTER
DUPLEX REGISTER
IN914'S
+10V
LEVEL CONVERSION
F910959
2.2KΩ
PULL-UP RESISTORS
STROBE
DECADE ELECT CODE LATCH
MC7475P
STROBE
MC7475P
INTERVAL SELECT CODE LATCH
DECADE FREQUENCY MULTIPLEXER
STRB A B C
SN74151
INPUTS OUT
STRB A B C
SN74151
INPUTS OUT
INTERVAL FREQUENCY MULTIPLEXER
÷1
PROGRAMMED CLOCK OUTPUT
DECADE FREQUENCY
F930059
÷2
DECADE DIVIDERS
÷2
F930 059
3 INPUT NAND GATES
÷5
10 MHz CRYSTAL OSCILLATOR
F930059
÷5

Device No.	Manufacturer	Description
F910959	Fairchild	Hex inverter-six level converters, high level to +4 to +20 V.
MC7475P	Motorola	Quad latch (4 bistable latches)
SN74151	Texas Instrument	Data selector/multiplexer
F930059	Fairchild	4-bit shift register
F900359	Fairchild	Triple 3-input NAND gate

Fig. 6. Schematic diagram for PRTB programmable clock.

grammed output, each decade output from 1 MHz to 0.001 Hz is available externally by patchboard connection.

Also available on the data acquisition panel were an ENCODE output bit which could be set and cleared by the computer, and a FLAG input terminal to allow external setting of the FLAG bit on the computer I/O channel used for data acquisition. The end-of-conversion (EOC) flip-flop of the ADC was normally connected to the FLAG input; however, it was possible to connect any appropriate externally generated signal to the FLAG input. Six bits of digital information could be transferred to or from the computer through the data acquisition panel using patchboard connections on the panel. Other generally useful functions available on the data

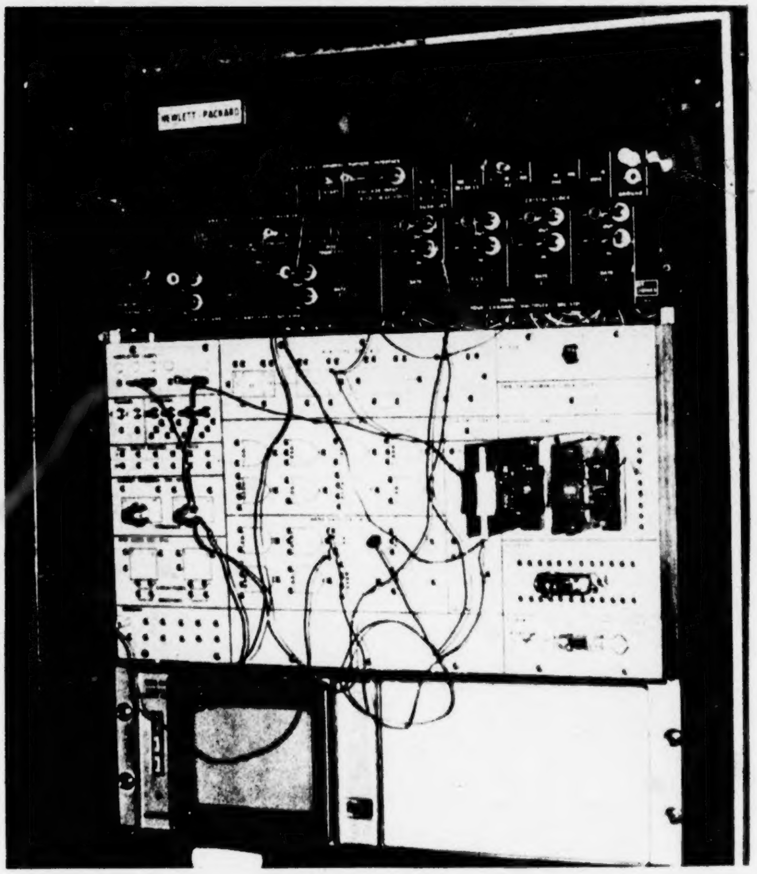


Fig. 7. View of general-purpose data acquisition and interfacing panels for PRTB system.

acquisition panel included patchboard connected to various logical devices such as 16 AND, OR, and NOT gates, four flip-flops, two one-shots, four analog switches, four relay drivers, a Schmitt trigger, a track-and-hold amplifier, six indicator lamps, and two push button switches with Schmitt trigger outputs. Figure 7 provides a view of the general-purpose data acquisition and interface panel used by the students. [See Malmstadt and Enke (1969) for a general discussion of characteristics of control logic nodules.] DEC R-series logic (Digital Logic Handbook, 1968). Flip-Chip cards and power supplies were used for logic functions. For interface design which required more sophistication or more logic elements than could be obtained on the data acquisition panel, a DEC patchboard R-series Logic Lab (Digital Logic Handbook, 1968) was available. Most experiments did not require interface hardware beyond that available on the data acquisition panel. All input and output to the computer were buffered with level conversion devices on the H.-P. 2115A system so that all external connections are compatible with DEC R-series positive logic (Digital Logic Handbook, 1968). DEC W601 and W510 level conversion cards were used.

An interfaced digital I/O module was also available for general-purpose digital communication between computer and experiment.

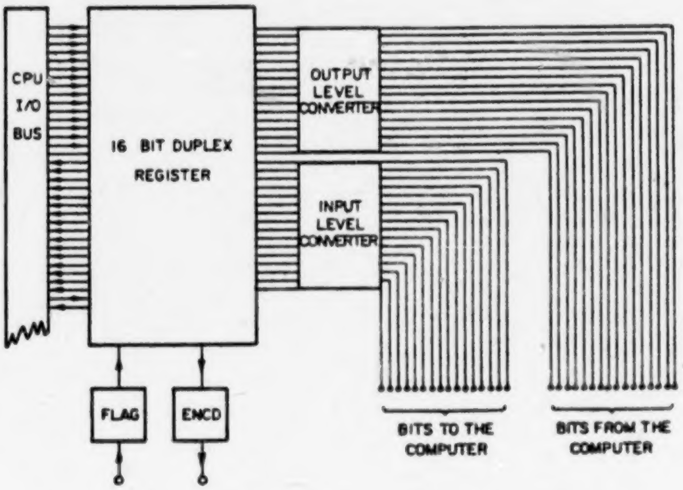


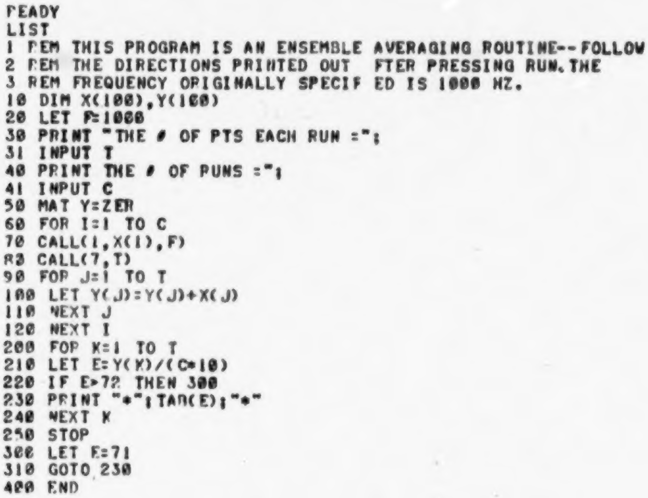
Fig. 8. Digital I/O interface module for PRTB.

The characteristics of this module are illustrated in Fig. 8. This digital I/O module was connected to a different I/O channel than the data acquisition panel. Thus, a separate independent ENCODE output and FLAG input were available through this module. The use of the digital I/O capability has been described above.

D. Results and Discussion

1. *Evaluation of the PRTB System*

For the purpose of illustrating the capabilities of the PRTB software, the laboratory computer systems and software were tested with a synthetically generated experimental output which simulated typical experimental data. The illustration involved application of ensemble averaging to the analysis of an experimental output which provided a repetitive voltage waveform with superimposed large amplitude random noise. The waveform for the ensemble averaging experiment had a fundamental frequency the order of 10 Hz, with superimposed large amplitude random noise. The experiment was



```
FEADY
LIST
1 FEM THIS PROGRAM IS AN ENSEMBLE AVERAGING ROUTINE--FOLLOW
2 FEM THE DIRECTIONS PRINTED OUT  FTER PRESSING RUN.THE
3 REM FREQUENCY OPIGINALLY SPECIF ED IS 1000 HZ.
10 DIM X(100),Y(100)
20 LET F=1000
30 PRINT "THE # OF PTS EACH RUN =";
31 INPUT T
40 PRINT THE # OF PUNS =";
41 INPUT C
50 MAT Y=ZER
60 FOR I=1 TO C
70 CALL(1,X(I),F)
R3 CALL(7,T)
90 FOP J=1 TO T
100 LET Y(J)=Y(J)+X(J)
110 NEXT J
120 NEXT I
200 FOP K=1 TO T
210 LET E=Y(K)/(C*10)
220 IF E>72 THEN 300
230 PRINT "*";TAB(E);"*"
240 NEXT K
250 STOP
300 LET E=71
310 GOTO 230
400 END
```

Fig. 9. Program for ensemble averaging experiment.

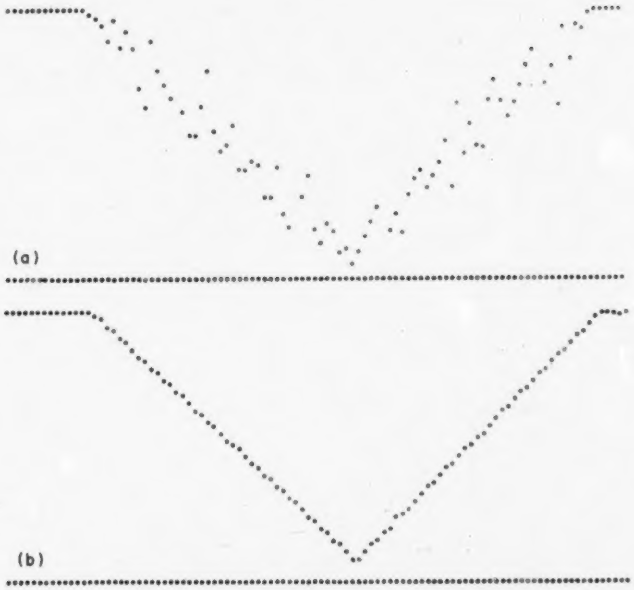


Fig. 10. Data from ensemble averaging experiment, plotted on teletype. (a) Original data (single cycle). (b) Data after 100 averaging cycles.

repeatable and could be triggered by computer output. The programming objectives included repetitive initiation of the experiment, synchronization of data acquisition wth experimental output, and repetitive coherent summation of the digitized waveforms to accomplish ensemble averaging. Finally, when the experiment was completed, the averaged data were normalized and plotted on the teletype terminal. The computer program used is given in Fig. 9. Figure 10 shows the original output waveform and the results of ensemble averaging.

The above experiment is executed by students in the laboratory. Other experiments include data acquisition and processing with synthetic experimental signals as described in Section I; gas chromatography (Perone and Eagleston, 1971); kinetic enzymatic analysis; differential kinetic analysis; and computer-controlled spectrophotometry. These and other experiments will be described in detail elsewhere (Klopfenstein *et al.*, 1972).

2. *Observations*

It can be seen from the above examples that actual implementation of the PRTB system is relatively straightforward. Moreover, for all its simplicity, considerable experimental measurement capability exists in the system.

The requirements for implementing the PRTB system in the undergraduate laboratory include the following.

(1) An introduction to the BASIC language for computational purposes. This requires 1 to 2 hours of lecture time. It also requires that students have access to the laboratory computer (or other computer facilities providing BASIC capability) during the first few weeks of the semester to handle homework and laboratory computations.

(2) An introduction to the fundamental concepts of on-line computer operation. This requires about 6 hours of lecture. These introduce the student to the whole field of on-line computer applications and the technological details with which they must be familiar to implement this approach in the laboratory. These details include an introduction to the fundamentals of digital logic, timing, and synchronization. This involves the discussion of simple gates (AND, OR, NOT), the flip-flop, the one-shot, and the analog switch. The functional characteristics of these devices, as well as of analog-to-digital and digital-to-analog converters and voltage amplifiers, for interface design are defined. No attempt is made to provide a rigorous understanding of the electronic principles or detailed circuitry for these devices. The emphasis is on the student's being able to recognize and use the fundamental characteristics of these devices for interface design.

(3) Finally, the actual laboratory assignments are carried out. These assignments are designed to introduce the student to the analytical technique and methodology, just as would be done in the absence of the laboratory computer instrumentation. In addition, the student is expected to utilize the on-line computer as a data acquisition and data processing tool.

The final question to be considered here is what specific benefits are derived by the student from the use of the laboratory computer in on-line experimentation. First of all, the student is obviously exposed to state-of-the-art technology in laboratory experimentation. Second, the student is encouraged to use more rigorous data processing approaches than previously feasible for laboratory assignments. Third, the student becomes keenly aware of the factors that limit and define experimental accuracy and precision. (This is a direct result of the fact that the computerized experiments can be re-run conveniently with modified parameters, and the results can be evaluated rapidly with the laboratory computer.) Fourth, the student is able to make a first-hand comparison of the effectiveness of conventional and computerized laboratory methodology. And, finally, the student's interest in the science of quantitative chemical experimentation is strongly stimulated.

Acknowledgments

The contributions over several years of the National Science Foundation for the development of the short course and the graduate and undergraduate educational programs are gratefully acknowledged. Pertinent NSF grants include GP-8528, GP-8677, GJ-190, GJ-403, and GJ-428.

The author also wishes to acknowledge the invaluable assistance of Dr. David O. Jones who provided the design concepts for instructional interface modules, Professors H. L. Pardue and Stuart P. Cram who have taught the digital logic section of the short course, and Professor J. W. Amy who has provided instrumentation support and advice.

The author also wishes to thank Professors L. B. Rogers and F. E. Lytle for their respective contributions to the suggestion, encouragement, criticism, and technical support of these programs. Finally, the author wishes to acknowledge Mr. John F. Eagleston who did most of the developmental work in generating a successful PRTB system.

References

1. "Digital Logic Handbook." (1968). Digital Equipment Corporation, Maynard, Mass.
2. Kemeny, J. G., and Kurtz, T. E. (1967). "BASIC Programming." Wiley, New York.
3. Perone, S. P., and Eagleston, J. F. (1971). *J. Chem. Ed.* **48**, 438.
4. Malmstadt, H. V., and Enke, C. G. (1969). "Digital Electronics for Scientists." Benjamin, New York.
5. Perone, S. P. (1970). *J. Chem. Ed.* **47**, 105.
6. Perone, S. P., and Eagleston, J. F. (1971). *J. Chem. Ed.* **48**, 317.
7. Klopfenstein, C. E., Wilkins, C. L., and Perone, S. P. (1972). In preparation.

S. P. Perone
Purdue University
Lafayette, Indiana 47907

A Laboratory Course on Digital Computers in Chemical Instrumentation

The chemist today finds himself in the midst of a dramatic revolution in scientific instrumentation. Digital electronics and laboratory-scale digital computers appear destined soon to dominate the whole area of chemical experimentation. This revolution has come about so abruptly that the practicing scientist recognizes a very real technological gap in his background. He realizes that computerized instrumentation can provide benefits in routine and research work—but he doesn't know how to get started. To compound his frustration, he even finds it difficult to communicate with the experts and computer manufacturers. Not only is the *jargon* of digital computer technology quite foreign to the uninitiated, but the fundamental concepts of digital instrumentation are basically unfamiliar. The scientist is accustomed to thinking of experimental data and instrumentation from an *analog* viewpoint, and the introduction to the digital world is not without some difficulty. Not that the material is difficult—it is really quite simple for the scientist trained in logic—but mastery of the material necessitates the generation of drastically different instrumental and experimental concepts than the scientist is generally accustomed to. The principles, methodology, and jargon of digital instrumentation are so different from those of analog instrumentation that even the instrument-oriented scientist has difficulty mastering the area independently. Moreover, the scientist's ability to use the large data processing computer is of little help in understanding *on-line* applications of the digital computer.

Because scientists have recognized this technological gap, many have taken steps to overcome it. This has required herculean efforts on their part, generally, since very little detailed information is available. Most of what is available is provided by computer manufacturers, and may require thorough familiarization with a particular computer for any appreciation. The most fundamental problem, however, is that an adequate appreciation of this new technology can not be acquired through reading or attendance at technical meetings. Moreover, it is not sufficient to have a computer available in the laboratory which the scientist can play with in his spare time. Nor is it sufficient to develop programming skills which are not oriented toward on-line experimental applications. The most efficient way to develop the technological skills necessary for the incorporation of the digital computer in laboratory experimentation is by total involvement in an intensive training program which includes substantial "hands-on" experience with digital instrumentation and the digital computer in an experimental on-line environment.

Thus, a need has existed for some time now to provide an educational program concerned with digital computer instrumentation for the chemistry laboratory. This type program is urgently sought by the practicing scientist—i.e., one who has completed his formal education. Moreover, this type program should be particularly valuable to our future scientists—the graduate and undergraduate students in our colleges and universities. Such a program has been instituted in the Chemistry Department at Purdue University under the direction of this author. It is the purpose of this article to describe the basic philosophy of this program as well as to provide the reader with many of the details of its operation. In addition, an evaluation of the results of this program will be presented.

On-Line Computerized Experimentation in the Laboratory

To put this entire presentation in perspective, it is necessary to provide some background regarding on-line digital computer technology. A common characteristic of the laboratory-scale digital computer is a relatively small core memory, usually providing 4K or 8K of random access storage of digital data and programs. A memory of this size, however, can be more than adequate for laboratory control operations, and, with careful programming, some fairly sophisticated data processing.

A small digital computer may have an *instruction set* which includes the order of 50–100 *machine language* instructions. Each of these instructions corresponds to a specific binary coding which, when decoded by the computer, results in the execution of a fairly simple arithmetic or logical step. Examples of some simple machine operations are: binary addition of a datum in some memory location to the contents of an arithmetic register (accumulator); transfer of the contents of an accumulator to a memory location (and vice-versa); rotation of the binary digits ("bits") of the accumulator contents to the left or right; and the application of logical tests such as determining if the accumulator is zero, non-zero, odd, even, positive, negative, etc.

Obviously, the repertoire of machine instructions includes some rather elementary operations. However, by developing appropriate "programs" composed of many of these elementary operations, the most sophisticated mathematical computations can be carried out. Since the computer can execute instructions so rapidly—the order of 10^5–10^6 instructions per second—it can complete complex computations with fantastic speed. For example, a program to multiply two *n*-bit integer values might require a program including 50 or 60 statements, but may require only about 200 μsec for comple-

[Reprinted from Journal of Chemical Education, Vol. 47, Page 105, February, 1970.]
Copyright, 1970, by Division of Chemical Education, American Chemical Society, and reprinted by permission of the copyright owner

tion. Thus, some 5000 multiplications per second could be accomplished.

Thus, the digital computer comes on as a very simple-minded device, which must be told how to accomplish even the most fundamental computations, but which can accomplish these operations with blinding speed. Moreover, it is a tireless machine which will be content to calculate endlessly and consistently. It is also a very versatile device, since it is programmable and capable of accomplishing an infinite variety of computational, logical, or control operations. Finally, it is a device which can (in fact, *must*) communicate in a variety of ways with the outside world. It is this characteristic which defines the computer as a general-purpose experimental device.

Off-Line Computers

The computer configuration with which most scientists are familiar is the *off-line* system. This configuration is diagrammed in Figure 1. To use the computer in this configuration, the scientist typically will write a data processing program in Fortran or some other higher-level computer language, run the experiment(s), manually tabulate the data from strip chart recorder or oscilloscope trace, transfer the tabulated data to punched cards, add the data cards to the deck of program cards, transport the combined card deck to the computer center for processing, and then wait until the program has been executed and the results printed. Turn-around times may vary from a few minutes to a few days, depending on the capacity of the computer facility and the number of users.

There are many variations of the above description of off-line computer usage. For example, the experimental system may include automatic data acquisition and digitizing devices which might store data on punched card, punched tape, or magnetic tape buffers which can then be transmitted more conveniently to the remote computer center. Alternatively, the laboratory may be equipped with a remote terminal (such as a Teletype) through which the investigator may enter his programs and data, as well as receive a print-out of results from the computer.

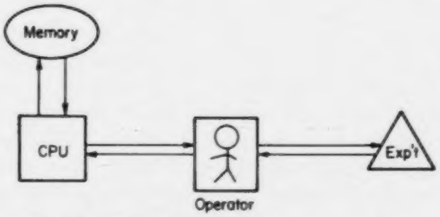


Figure 1. Typical off-line computer configuration.

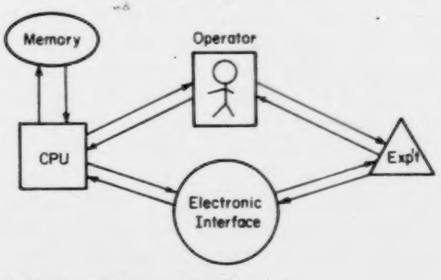


Figure 2. Typical on-line computer configuration.

The important common characteristic of all off-line computer systems, however, is that the experimental data are transmitted to the computer through some intermediate storage medium, and they are processed after some finite time delay has occurred. Depending on the modes of data acquisition and transmission to the computer, the turn-around time of the computer facility, and the speed with which the investigator can interpret the result print-out, the time delay for experimental modifications based on the results of previous experiments can be excessive. Should this *reaction time* be a critical factor, an off-line computer facility may be inadequate for the particular experimental studies.

On-Line Computers

For the investigator who requires very rapid or instantaneous results from his computer system—for whatever reason—the solution may be to employ an on-line computer system. Figure 2 presents a block diagram for a typical on-line computer configuration. The most important distinction of this configuration is that there is a *direct* line of communication between the experiment and the computer. The line of communication is through an electronic interface. Data are acquired under computer control or supervision, and the program for data processing is either resident in memory or immediately readable into memory to provide for very rapid completion of the computational tasks. Results may be made available to the investigator quickly by means of teletype, line-printer, oscilloscope, or other form of print-out. In addition, the computer may be programmed to communicate directly with the experiment by controlling electronic or electro-mechanical devices—such as solid state switches, relays, stepping motors, servo motors—or any other devices which can be activated by voltage level changes.

The advantages of on-line computer operation can be summarized as the following: *Elimination of the middle man*, and the concomitant substitution of an electronic interface between the computer and the experimental system, which is much more compatible with the computer's characteristics and which does not suffer from the inherent inadequacies of the human as a communication link. For example, the computer can accept input data at rates of the order of 10^5–10^6 per second, and can instantaneously transmit control information or commands back to the experimental system. If this line of communication were handled through the human investigator, as in the typical off-line configuration, the response time of the overall experimental and data processing system would be many orders of magnitude slower. Moreover, the electronic communication link is generally a more reliable, tireless, objective, and accurate information transfer medium.

The possibility of *direct computer control of the experiment* is a second advantage associated with on-line computer operation. This facility allows the complex logical and decision-making capabilities of the computer to be implemented for an infinite variety of laboratory automation or experimental design problems.

A third advantage of the on-line computer is that *real-time interaction between computer and experiment* is possible. That is, because the computer can make computations and decisions at speeds exceeding most ordinary data acquisition rates, it is possible for the computer to execute experimental control modifications before a given experiment has reached completion. These "real-time" operations allow the possibility of using the computer to monitor the progress of an experiment and modify that experiment to follow a pathway which provides more optimum conditions and which could only have been arrived at as a result of some observation of the initial course of the experiment.

A *breakdown of the logistical barriers of the remote computer system* is an additional advantage of on-line computer operation. Because of the instantaneous, direct communication link between the experiment and the computer, the physical location of the computer with respect to the laboratory is relatively unimportant. Moreover, the mechanical and logistical roadblocks typically imposed by the computer facility toward the off-line introduction of data are irrelevant.

One very important aspect of using computers for on-line experimental work that should be emphasized here, however, is that some, if not all, of the programming must be done in machine—or assembly—language. That is, higher level languages—such as Fortran—do not have the capability for on-line operation with experimental systems. Moreover, higher-level languages tend to be very inefficient in the use of memory space and in execution time. Both of these factors are critical when small computers are used for on-line work. Thus, the scientist wanting to use the small digital computer in the laboratory must become competent in programming in the fundamental machine language—a formidable task.

An Educational Program in Computerized Instrumentation

It can be seen from the preceding discussion that the widespread use of digital computers on-line in chemistry laboratories will certainly revolutionize routine laboratory methodology. Today's scientist must learn to live with the digital computer and to use it profitably for laboratory work. On this premise we have developed at Purdue an intensive 3-wk summer short course entitled "Digital Computers in Chemical Instrumentation." The course is designed for practicing scientists in colleges, universities, industrial laboratories, government laboratories, and hospitals, who would like to be introduced to digital instrumentation and the technology of digital computer implementation in the laboratory. It is the details of this summer program which will be discussed here.

Objectives

The purpose of the 3-wk intensive course developed at Purdue was to afford participants an opportunity to acquire a familiarity with digital instrumentation and digital computers in the laboratory. The course was designed such that students could develop, in a short period of time, the ability to write machine-language programs, to utilize the computer for typical laboratory input/output functions, and to design and implement software and hardware for interfacing real experimental systems to the computer for on-line operation. An essential feature of the course had to be the availability of several digital computers and many digital logic modules to provide a large amount of hands-on laboratory experience for each participant. To make the experience most worthwhile to the participants the lecture and laboratory material were developed within the context of applications in chemistry and related sciences. (The course material was based on the assumption that participants had no previous experience with computers, digital logic, or electronics, but were generally familiar with chemical instrumentation and laboratory methodology.)

Another important objective of the summer course project was to use it as a means of developing the lecture and laboratory ingredients, as well as the staff, for undergraduate and graduate courses at Purdue incorporating computers in chemical instrumentation. Moreover, many of the university and college professors taking the summer course would be encouraged to develop computer-oriented course work at their respective institutions. Thus, by providing the summer course for educators the development of curricula which includes the on-line implementation of computers in undergraduate and graduate scientific work might be fostered in a most efficient manner.

Approach

The basic philosophy of the 3-wk summer course is simply to share with the participants the background, technique, and experience in computer instrumentation developed at Purdue and elsewhere by the faculty and graduate instructors of the course. For the participants to acquire the desired skills and background in the 3-wk period, it is absolutely necessary to provide for as much individual access to computing equipment and digital instrumentation as possible and practicable. To provide this "hands-on" access to the equipment in the most efficient manner, a great deal of preparation outside the laboratory is required. Thus, off-line teletypes are provided for program preparation and editing; labs are open and supervised on evenings and Saturdays; and a $1^1/_2$-hr lab lecture is presented each day. Most importantly, a large ratio of teaching staff to students is provided, so that students can obtain help with a minimum of delay. At least one instructor for every four students in lab is the normal working ratio at all times.

The significance of the emphasis put on the efficient use of the available computing equipment can not be overemphasized. The natural tendency of the novice programmer is to sit down at the computer console and try to compose a program. The students in our course learn very quickly that they must have their programs fully composed and punched on tape—with perhaps one or two alternates, in case the first doesn't work—before they even get to the computer console. When programs do not execute properly, they are encouraged to spend a minimum of formal lab time in time-consuming de-bugging operations, and to restrict their de-bugging efforts to spare time outside of lab, in the evening, or on Saturdays. Thus, when a program is to be edited and reexecuted by the computer, a minimum of "hands-on" time will be consumed by the student trying to reason away the logical errors in his program.

Another important aspect of the course is that it is not limited to the concepts, principles, and techniques associated with the "small" digital computer. The formal lecture material discusses large, time-shared systems as well as small, dedicated systems. Moreover, one laboratory assignment involves writing a program and designing an interface to time-share several peripheral devices. The principles learned in this laboratory assignment are extended in lecture to describe the operation of larger time-shared computer systems.

Requirements

Equipment. An essential requirement for the presentation of the short course on computers in chemical instrumentation is the availability of sufficient hardware to allow adequate "hands-on" access by each participant. The basic equipment available for a 12-man laboratory session includes: three Hewlett-Packard digital computers (one, 2116A; two, 2115A) with 8K-word core memories, ASR/33 teletypes, and high-speed paper tape readers. One of these computers is equipped with a high-speed paper tape punch. Two of the computers are equipped with hardware multiply/divide capability, and one has a two-channel direct-memory-access capability. In addition, each computer is completely equipped for high- or low-speed computer-controlled digital data acquisition with three 10-bit ADC's (each capable of data acquisition rates up to 20 or 30 KHz), and three frequency-selectable, gated clocks for precise timing and synchronization of data acquisition and other experimental events. Two computers are equipped with high-precision (16-bit), high-speed (20 μsec rise-time) D2A converters. Two computers are equipped with oscilloscope display units interfaced to the computer through dual eight-bit D2A converter modules. (One computer also has a digital plotter and a seven-track, 556 bpi, digital magnetic tape unit available. However, these items of equipment have not yet been used in student assignments.)

In addition to the computer equipment, each 12-man laboratory has available six patchboard digital logic training stations (Digital Logic Laboratories, Digital Equipment Corp, Maynard, Mass.) and six dual-trace wide band-pass oscilloscopes for the laboratory assignments on digital logic. These units are also used for assignments requiring interface design for computer-interfaced instrumentation.

Other items necessary to completely equip the laboratory include a variety of specially-designed patchboard modules and associated electronics for instrumentation interface design, which are described in detail below. Also, at least three off-line ASR/33 teletypes are provided for program preparation on punched paper tape.

Personnel. One of the most subtle, and yet most critical requirements for presentation of the course described here is the availability of qualified individuals who understand the software and hardware aspects of computerized experimentation in the laboratory, and who can teach the material. We are indeed fortunate here at Purdue in that we currently have a relative abundance of such personnel.

The explanation for the presence of qualified personnel here is twofold. We have several staff members who have become actively involved in on-line computer applications in their research, and we offer a formal one-semester course for graduate students designed to provide first-hand familiarity with on-line computer techniques in the chemistry laboratory. (This graduate-level course—first offered informally in Spring, 1968—formed the basis for the present intensive 3-wk summer course.) Thus, by offering both the graduate course and the 3-wk summer course, we have developed a unique group of well-trained teaching assistants.

Detailed Course Description

Schedule

The summer course spans the last three weeks in June. The first day (Sunday) is devoted to orientation and the first lab lecture. The next 13 weekdays are devoted to a rigorous schedule of laboratories, formal lectures, lab lectures, and open lab time. Each participant spends $1^1/_2$ hr in lab lecture, $1^1/_2$ hr at formal lecture, and $2^1/_2$ hr in formal laboratory between 8 A.M. and 9 P.M. daily. In addition, the laboratories are open and supervised from 9 P.M. to midnight each evening so that students might work ahead, catch up, or simply engage in an informal discussion with some of the staff. Labs also are open and supervised on Saturday from 8 A.M. to 5 P.M. One day is devoted entirely to the demonstration of various on-line systems operating in the Chemistry Department at Purdue (including mass spectrometry, electrochemistry, flash photolysis, gas chromatography, and stopped-flow kinetic spectrophotometry). The last day is devoted to a "mini-symposium" on computer applications in chemistry, to which several expert outside speakers are invited.

Content

The formal lecture material covers the following three areas: digital logic and digital instrumentation; computer programming for on-line laboratory applications; and hardware and software design for instrumental interfacing, time-sharing, and computer control in the laboratory. A detailed lecture outline for the 1969 summer course is given in Table 1.

The formal laboratory assignments, likewise, were broken down into three categories: basic digital logic design and operation (4 da); basics of assembly language computer programming and operation (4 da); and interface design and computer programming for on-line applications (5 da). Lab lectures ($1^1/_2$ hr per day) were designed to prepare the student in advance for each

Table 1. Short-Course Lecture Schedule, Summer 1969

I. Introductory Remarks
II. Elements of Digital Logic
III. Digital Functional Units
IV. Number Systems; Binary Arithmetic, etc.
V. Boolean Algebra—Applications to Digital Circuitry
VI. Analog-to-Digital, Digital-to-Analog Conversion
VII. Computer Organization and Machine Language
VIII. Assembly Language Programming; Simple Arithmetic Routines
IX. Chemical Data Processing Routines
X. Programming the Computer for Input/Output with Standard Peripheral Devices
XI. Applications of Digital Logic in Chemical Instrumentation
XII. Digital Logic in Experimental Timing, Control, and Synchronization for Computer Interfacing
XIII. Data Acquisition and Real-Time Computer Control in Experimental Systems
XIV. Time-Sharing Systems; Combining Languages; Large Computer Systems

Table 2. Outline of Laboratory Assignments, Summer 1969

- A. Digital Logic
  - I. Flip-flops and counters
  - II. Gates and fundamental logic circuits
  - III. Boolean equations and logic design
  - IV. Digital control, timing, and synchronization
- B. Computer Programming Fundamentals
  - I. Elementary machine—and assembly—language programming
  - II. Simple arithmetic programs
  - III. Chemical data processing algorithms
  - IV. Input/output programming
- C. Interfacing and On-Line Computer Programming
  - I. Data acquisition from a transient experiment
  - II. Digital input/output
  - III. Data acquisition and processing of gas chromatographic data
  - IV. Ensemble averaging
  - V. Time-shared service of several peripheral systems
  - VI. Multiplexing and digital-to-analog operations

laboratory assignment. Table 2 contains a detailed list of laboratory assignments.

Laboratory Procedures and Organization

The organization of the laboratory and lab assignments is certainly most critical to the successful presentation of the short course.

With the available equipment providing the basic limitation on laboratory size, the labs were organized to make most efficient use of the hardware. This was accomplished by, first, dividing each lab into two groups: One-half start out working with the computer equipment; one-half start out working with digital logic equipment. Four days are required to complete the basic assignments in each group. Then, the two groups of students exchange equipment, and the basic set of assignments are repeated during the next four days. Two students are assigned to one computer per lab; one student is assigned to one logic station per lab.

While the basic sets of laboratory assignments are being carried out the logic group attends lectures II–VI (Table 1), while the computer group attends lectures VII–X. These basic lectures are primarily oriented towards the assigned laboratory material and are repeated when the two groups switch assignments.

After all participants have completed the basic sets of logic and computer programming assignments, five days of laboratory work follow which are oriented towards the development of programs and interfacing for on-line experimental applications. Students continue to work in pairs on these assignments using any of the hardware available in the lab to execute each assignment. The apportionment of the computer time must be carefully supervised by the lab instructors during this period so that the equipment is used efficiently. One approach found effective here was to make available a fourth computer system which was used only for assembling and editing of program tapes. This freed the other computer systems for program execution and experimental design.

An interesting innovation was introduced during the summer, 1969, course. Through the generosity of several computer manufacturers (Hewlett-Packard Co., Palo Alto, Calif.; Digital Equipment Corp., Maynard, Mass.; and Varian Associates, Palo Alto, Calif.) several different computer systems were made available during the last week of the course so that students might have the option of working with alternative equipment. The only requirement imposed was that they complete all normal course assignments up through at least the basic set of on-line experiments (C-I, C-II, Table 2). Thus, confident that they had been exposed to all the basic technology we would hope to provide, we could turn students loose on unfamiliar equipment. Our objective was to prove to students that the operational differences between systems are not so great as they appear once the fundamental principles have been mastered.

To make this possible in 1969, the additional equipment made available included two DEC PDP-8L computers with high-speed paper tape input/output, interfaced 12-bit ADC and four-channel multiplexer, and ASR/33 teletype; two DEC PDP-12 computers with full standard interfaced experimental input/output facilities (ADC, D2A, etc.), two LINC tape decks, oscilloscope display, and ASR/33 teletype; a Varian 620-i computer with provision for experimental I/O, oscilloscope display, and an ASR/33 teletype. In addition, two Hewlett-Packard computers (2116B and 2115A) with high-speed paper tape input and ASR/33 teletype were provided by the manufacturer for use along with the University-owned Hewlett-Packard systems.

The results of using the alternative computer equipment were very gratifying. Nearly all of the students who switched to the unfamiliar equipment were able to quickly master the new assembly language software and operating procedures. In fact, many were able to develop programming and interfacing for on-line operation. All this was accomplished with only a total of three days exposure to the equipment! These results seemed to prove that the necessity for exposure to one type of computer equipment in the laboratory part of the course is certainly no hindrance to the later use of other systems.

Hardware for On-Line Experimentation and Interface Design

Because it was unrealistic to assume that students would be skilled in electronics, the development of lecture and laboratory material on interfacing experimental systems for on-line operation was particularly challenging. However, by building on the fundamentals of digital logic learned in the early part of the course, it is possible to cover adequately the topic of interface design.

The basic approach used involved recognizing that the digital computer communicates with the outside world by the execution of instructions which either cause binary voltage level changes to occur at specific external terminals, or which recognize voltage level changes effected by external devices. This understanding, coupled with the fact that the digital computer can neither accept nor output normal analog (continuously variable) signals, determine the type of interface functions required.

One primary group of interface elements to be considered are *translational* modules, such as analog-to-digital converters and digital-to-analog converters. In addition, such devices as level converters and analog amplifiers contribute to translational functions. Also important for consideration are multiplexers and signal transmission problems.

Those elements essential to proper communications are the logic, timing, and control components. These include flip-flops, gates, one-shot delays, Schmitt triggers, clocks, counters, analog switches, and electromechanical devices like relays and stepping motors. The student is made aware of the functional characteristics of all of these interface elements. (Many are actually used and tested individually in the laboratory.) He is then responsible for designing interfaces for assigned experiments by selection and combination of appropriate elements to provide both the translational and logical functions required. This means that he must be able to provide for the coordination and synchronization of the computer's program execution with the experiment and with the data acquisition system.

To implement these principles in the laboratory, we provide students with the building blocks with which the actual interface can be constructed in patchboard fashion, once the logical design is complete. Included in the patchboard interfacing package are: a 10-bit (30 KHz) externally-controlled ADC, a gated 10 MHz crystal clock with scaled outputs available in decade steps from 1 MHz to 0.01 Hz; a four-channel multiplexer; two sample/hold amplifiers; and 8-bits of binary control and logic I/O. Another type of patchboard interface panel is available which provides for 16-bit digital I/O only. The DEC logic labs are available to allow incorporation of logic and timing elements (gating, time-delays, etc.) not included in the basic interface panel. All interconnections can be made externally with plug-in leads. Figure 3 shows the basic interfacing equipment. Figure 4 shows the equipment in operation.

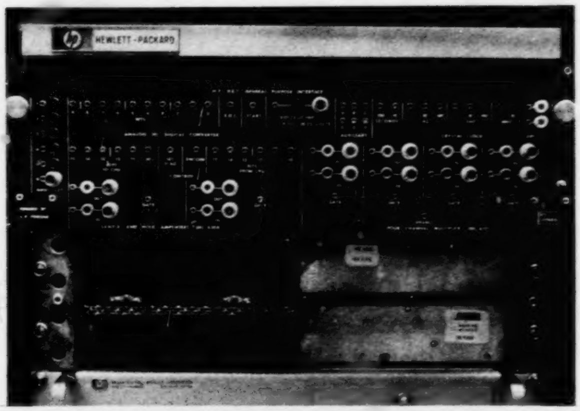


Figure 3. Basic interfacing module.

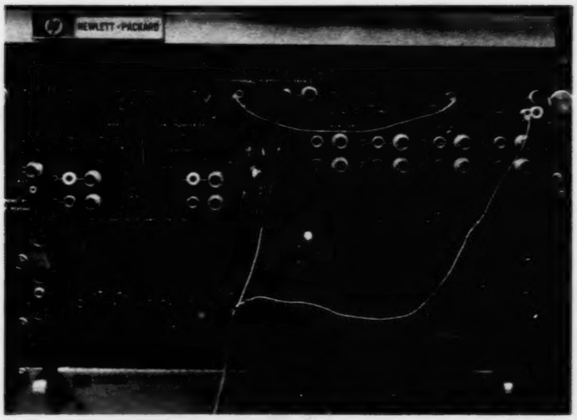


Figure 4. Basic interfacing module wired for specific on-line experiment.

On-Line Experiments

The ultimate objective of the laboratory work is for students to be able to carry out experiments on-line with the digital computer. Thus, the final set of laboratory assignments require the student to design and implement the programming and interfacing for a variety of on-line experimental systems. Only the detailed characteristics and data acquisition requirements for each experiment are specified for each assignment.

Basically, two types of experimental systems were provided for laboratory assignments: synthetic and real. The synthetic systems were electronically-simulated experimental outputs where the control, synchronization, and signal-handling problems were made nearly identical to several real systems. These electronic simulators were constructed on printed-circuit cards which could be plugged into the DEC Logic Labs (see Fig. 5). Thus, patchboard interfacing could be implemented easily.

Simulated experimental output included: an externally-triggered rapid exponential transient decay voltage (time constant of 0.1 sec); an externally-triggered repetitive voltage wave form with superimposed large amplitude random noise for signal-averaging studies; simultaneously-generated multiple voltage waveforms for multiplexing experiments; and a synthetic typical gas chromatographic output including several peaks of varying shape and size.

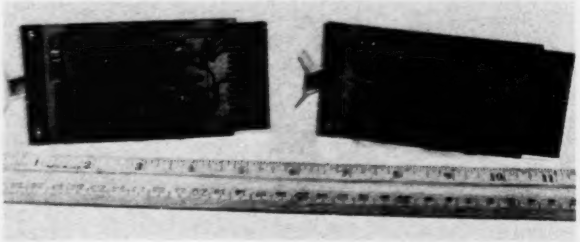


Figure 5. Typical plug-in synthetic experimental signal generators. *Left,* Provides triggered repetitive voltage wave-form (10Hz) with superimposed large amplitude random noise for ensemble averaging experiments. *Right,* Provides triggered simultaneous multiple voltage waveforms for multiplexing experiments.

The above simulated systems provided convenient, reliable, and realistic on-line experimental experience for each student. The most obvious advantage was the freedom from chemical and instrumental uncertainties invariably associated with real chemical laboratory systems. On the other hand, it was desired to provide some experience with real laboratory systems, since this appears to make the most lasting impression on students. Thus, laboratory gas chromatographs and a variety of sample mixtures were made available for students to apply their on-line data acquisition and processing skills. Appropriate amplification of the normal G.C. outputs could be provided by the selection of an operational amplifier-based plug-in module which could be mounted on a Logic Lab. The rest of the interface design was left to the student.

The minimum of assignments expected to be completed by all students included: data acquisition from a simulated transient experiment, using computer-controlled constant or variable data acquisition rates, and computer-controlled data acquisition from an external digital counter. Beyond these minimal assignments students could carry out ensemble averaging, multiplexing, and gas chromatographic experiments. In addition, assignments involving computer-controlled digital-to-analog waveform generation and time-sharing of multiple external systems could be executed.

Many of the above described assignments required that *real-time* data analysis be incorporated into the programming. That is, some assignments required data analysis during data acquisition. Mastery of this type of programming is indicative of an accomplished experimental programmer.

Results

The primary result of the development and presentation of the course described here has been to make available to practicing scientists, regardless of background or previous experience, a rapid, effective introduction to computer technology in the laboratory. In addition, the instrumentation and educational techniques developed for the summer course have been incorporated into a new graduate-level chemical instrumentation course now being offered at Purdue. Moreover, experiments have now been introduced into an undergraduate junior-level analytical course which utilize on-line computer operation.

Acknowledgment

The contributions of the National Science Foundation for the development of this course and the support of college teacher participation are gratefully acknowledged.

Pertinent NSF contracts include: GP-8528 and GP-8677. Also included are GJ-190, GJ-403, and GJ-428 from the Office of Computing Activities of NSF.

The author also wishes to acknowledge the invaluable assistance of Mr. David O. Jones, who provided the design concepts for instructional interface modules and supervised the development of laboratory experiments, Professor H. L. Pardue who has taught the digital logic section of the course, and Professor J. W. Amy who has provided instrumentation support and advice.

◆

S. P. Perone
and **J. F. Eagleston**
Purdue University
Lafayette, Indiana 47907

Introduction of Digital Computers into the Undergraduate Laboratory

Current trends in the chemical industry throughout the country indicate that the digital computer will soon become an indispensable part of chemical experimentation (*1–3*). It should be clear, then, that the chemist of the near future will be required to understand and use properly digital instrumentation and digital computers in the laboratory. Furthermore, educators at colleges and universities will need to develop undergraduate curricula providing the required background in computerized chemical instrumentation. In fact, the demand for this kind of training by practicing scientists has prompted the development of a summer short course here at Purdue for college teachers and others in chemistry and the biomedical sciences (*4*).

This paper describes the first steps in a long-range program involving the introduction of on-line computer applications in the undergraduate chemistry laboratory at Purdue. The introduction has been made as part of an introductory analytical chemistry course dealing with quantitative chemical measurements (Junior level at Purdue). A primary objective was that none of the chemical and instrumental principles currently developed in the course be eliminated, but rather that these be augmented by the incorporation of the on-line computer into many of the laboratory experiments. The manner in which such a program may be made feasible is described below.

Approach

It was our intention to introduce computer technology into the undergraduate chemistry laboratory without slighting the chemistry content. The approach has been to continue to include the same basic set of laboratory experiments—except that, eventually, many will be designed to require on-line communication with the digital computer. The student is required to understand and execute the laboratory experiment as usual—but, in addition, he is required to design the appropriate programming and communication elements for optimum interaction between the computer, experiment, and experimentalist. Thus, we envision the student's achievement to be one of *experimental design*, and the *proper utilization* of computing equipment in the solution of measurement problems in chemistry. Specific aspects emphasized include: the general-purpose nature of the digital computer (*i.e.*, how it can be applied to a large variety of experimental problems with modifications in programming); the computer as a control element in experimentation; the rapid response of the computer, allowing real-time interaction with experiments—with reaction times several orders of magnitude faster than the human operator; and the computer as an integral part of the chemical instrumentation.

To accomplish the above mentioned objectives, it was necessary to recognize, first of all, what could *not* be expected within the scope of this course: (1) we could not educate our students in the intricacies of machine-language programming, and (2) we could not provide them with the electronics background requisite for the sophisticated design and construction of interfacing between computer and experiment.

What *could* be done is to take advantage of students' previous exposure to off-line computer programming (with FORTRAN usually), which many—but not all students—have obtained before the Junior year at Purdue. We could also take advantage of their exposure to the fundamentals of electrical measurements in physics courses to instill them with the most elementary concepts of amplification, response, and noise. We could expect to provide them with the essential principles of digital logic required for interface *design*.

The most important requirement for the success of the program, however, was the development of two items: (1) a modified high-level programming language, which includes versatile data acquisition and experimental control sub-routines with conversational mode calling sequences; and (2) a general-purpose hardware package for interface design. This apparatus would provide for patch-board incorporation of those digital and analog modules required to complete the interface between computer and experiment. The general-purpose interface package requires only that the student be able to lay out the basic logic design, timing sequence, and the analog amplification or attenuation required.. He can then select appropriate pre-packaged plug-in units to implement his design. He does not have to tend to such details as level conversion, logic conversion, noise rejection, and other subtle design parameters. Yet, he will accomplish the most important part of the interface development—logic design.

Purdue Real-Time Basic System

The high-level programming language chosen for development in this work was the BASIC language (*5*). There were several reasons for selecting this language: (1) It is *easy to learn*, generally requiring the order of half a day exposure to develop a good working knowledge. (2) It is an algebraically-oriented *conversational* language. (3) It is *interactive*. That is, the compiler is *interpretive*, and therefore compiles and executes programs line-by-line. This allows programs to be entered, executed, and edited on-line thru a teletype terminal. It also provides for immediate turn-around and rapid error diagnostics. (4) BASIC is rapidly becoming a *universally acceptable* language. (5) BASIC is currently readily *available* at commercial time-share

[Reprinted from Journal of Chemical Education, Vol. 48, Page 317, May, 1971.]
Copyright, 1971, by Division of Chemical Education, American Chemical Society, and reprinted by permission of the copyright owner

terminals. This allows the convenient and economical learning of the language by large numbers of students for later on-line experimental applications.

It should be noted that other languages (e.g., FOCAL[1]) are available which have comparably desirable characteristics. Even languages such as FORTRAN or ALGOL which are generally considered strictly for off-line applications may be appropriate, particularly for systems with "load-and-go" capability.

The software system developed at Purdue will be referred to as "Purdue Real-Time Basic" (PRTB). The software includes, fundamentally, the BASIC compiler available from Hewlett-Packard Co., which manufactured the computers used in the undergraduate laboratory program (see Experimental Section). The modifications generated here at Purdue have involved the development of a series of machine-language subroutines which are directly callable from the BASIC software, and which are designed to communicate in a variety of ways with experimental systems. The PRTB data acquisition and control software are described below.

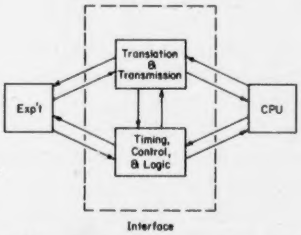


Figure 1. Schematic diagram of on-line computer configuration.

General Description of PRTB System

Figure 1 illustrates the general nature of an on-line computer system. On the one hand is located the laboratory experiment, the data from which the student desires to present to the computer for data processing. On the other hand is the digital computer itself, which is capable of high-speed programmed computational and control functions. In order to operate the communication link between the computer and the experimental system an electronic interface must be established. This interface accomplishes the functions of translation (analog-to-digital, digital-to-analog, decoding, logic conversions, etc.), timing, synchronization, and logical control. The programming language must take into account the nature of the electronic interfacing and the interfacing must be designed with the fundamental characteristics and capabilities of the programming language in mind. The following section describes the subroutines developed for on-line experimentation within the PRTB system. A summary of these subroutines and characteristics is given in Table 1.

Experimental Input/Output Subroutines for PRTB

Data Acquistion. Two different types of data acquisition subroutines are available within PRTB. They both work in conjunction with an external clock-controlled analog-to-digital converter (ADC). One is a subroutine (SB3) which should be called by the user's program whenever the computer should be waiting for the next data point to become available. When the next datum is digitized, SB3 takes the data point from the data acquisition device (ADC), converts the datum to floating point format, and stores it in the appropriate memory location for subsequent reference by the BASIC program. When SB3 is exited, the computer returns to the next statement in the BASIC program for execution. The new datum can be operated on, but, before the next data point can be acquired, SB3 must be called again.

A second type of data acquisition subroutine (SB7)

Table 1. Data Acquisition and Control Subroutines for PRTB

Subroutine	Function	Call Format
SB1	Initialize I/O. Set up data storage address and acquisition rate for other subroutines. "X(*I*)" data variable. "*F*" = data acquisition frequency.	CALL (1,X(I),F)
SB2	Used to start the CLOCK. The output control bit (ENCODE) can be connected to some external logic preceding the CLOCK ENABLE input. (NOTE: SB2 must be preceded by SB1.)	CALL (2)
SB3	Waits for ADC FLAG, indicating conversion completed. One datum is then taken from the ADC, converted to floating point, and saved as X(I).	CALL (3)
SB4	When called this subroutine waits for switch 0 of the console switch register to be set to a "1" before continuing with the next statement in the BASIC program.	CALL (4)
SB5	Outputs one control bit (ENCODE) and then waits for an event FLAG on the specified I/O channel, "C," before continuing. FLAG and ENCODE are cleared before exit.	CALL (5,C)
SB6	Waits for FLAGS on the ADC channel. When FLAG is set, the next statement in the BASIC program is executed. (Must be preceded by SB1 and SB2 if internal CLOCK is used to generate FLAGS.)	CALL (6)
SB7	Takes in complete block of data before returning to BASIC program. (Max. of 250 pts. Must be preceded by SB1 each time it is called.) "T" sets total No. of data pts. taken. Synchronized with start of experiment through external gating of the ENCODE output. Up to 20 KHz data acquisition rate possible.	CALL (7,T)
SB8	Causes output bit No. "Z" to be set TRUE on specified I/O channel, "C." Z,C specified in decimal. All other bits will be cleared.	CALL (8,Z,C)
SB9	Causes output of analog voltage through the DAC. "D" = mV output.	CALL (9,D)
SB10	Clears ADC channel. External Flip-flop driven by ENCODE will be cleared also. This can be used to turn off clock, stop experiment, etc.	CALL (10)
SB11	Causes output of binary voltage pattern, "Z," on specified output channel, "C." Binary pattern and channel specified in decimal.	CALL (11,Z,C)
SB12	Inputs status of 16-bit register on channel "C." The floating point equivalent is saved in memory as "Z."	CALL (12,Z,C)
SB13	Inputs status of a specific bit (No. "Z") on channel "C." Value of "S" is made "1" or "0" accordingly.	CALL (13,Z,S,C)

---

[1] Trade mark, Digital Equipment Corp., Maynard, Mass.

is one which allows for the acquisition of a complete block of data before exit. The external clock is started and synchronized with data acquisition within the subroutine. The basic difference between SB7 and SB3 is that SB3 is called to acquire one data point at a time and therefore allows for program statements to be executed during the time between acquired data points. Thus, an experimenter could devise a program which could process experimental data while the experiment was in progress and data were being acquired. This is referred to as "real-time" data processing. However, because the computer program must involve the relatively inefficient execution of BASIC statements between data points, there is a more severe limit on the speed with which data can be acquired without the computer getting out of synchronization with the experimental timing. In fact, SB3 is designed to detect when the computer has gotten out of synchronization; an error message will be typed, and the computer will halt. The user will then have to revise his program to require less real-time processing. With SB7, on the other hand, a complete block of data points is acquired before the subroutine is exited and control returned to BASIC. Therefore, the timing is limited by the efficiency of the machine-language programming developed within SB7. For our computer systems and software, it is possible to acquire data at rates as great as 20 KHz with SB7, but no real-time data processing is possible.

The limiting data rate when using SB3 is about 50 Hz, with a minimum of real-time data handling. If several BASIC computations are to be executed between data points when using SB3, the limiting data acquisition rate may be the order of 1 to 10 Hz. However, this is generally more than adequate for most experiments in the chemistry laboratory.

Experimental Control and Logic. In addition to data acquisition, there are several other ways in which the computer can communicate with the experimental system. One of these is through a digital-to-analog converter (DAC), where the digital output of the computer is converted by the DAC to an equivalent analog voltage level. One of the PRTB subroutines (SB9) provides the capability for driving the DAC, the output of which can be connected to external experiments. It is possible to generate a continuous voltage waveform output from the DAC by mathematical generation within BASIC of the discrete points making up the waveform and transmitting these through SB9 to the DAC in a repetitive fashion synchronized with an external clock.

Subroutines SB8 and SB11 allow the programmer to utilize specific output bits on a selected I/O channel to control external devices. There are 16 output bits available for this function. With SB8, the user can select which bit he wants to set by specification of the bit number. The setting of one of these output bits causes a corresponding binary voltage level change at the specified output terminal, and this can be used to close or open switches, start or stop experimental events, light indicator lamps, etc. SB11 allows the programmer to output a 16-bit binary voltage *pattern* with any simultaneous combination of "1" 's and "0" 's he chooses. This binary pattern is selected by including in the subroutine call the decimal equivalent of the binary number to be generated. Thus, more than one event can be controlled simultaneously.

SB12 and SB13 provide digital *input* information for program "sensing" of external situations. Thus, 16 binary voltage input terminals are available to the user, the status of which are acquired by either subroutine. SB12 transmits to the BASIC program the numerical equivalent of the input 16-bit binary voltage pattern. Because the status of these bits can be set by external events, the computer could use this information to make appropriate changes in the data processing or control programming.

SB13 is similar to SB12, except that only the status of a single specified input bit is determined. This subroutine is useful to the student because he need only specify a bit number in the call statement to check the status of a bit.

Timing and Synchronization. The most fundamental operation which must be accomplished by the PRTB system is the generation of a *time base* for all experimental functions. This is accomplished by the incorporation of a fixed frequency crystal clock (10 MHz) into the interface hardware. Also included is electronic count-down logic to scale the output clock pulses down to a useable frequency range for chemical experimentation (100 KHz to 0.01 Hz). The countdown logic can be modified under program control to select any frequency within this which can be generated by a decade-and/or a 1,2,5-countdown sequence. The programmed clock output pulse train is then available to control the timing on the ADC, DAC, or any other external hardware. Also available simultaneously are synchronous clock pulses representing frequencies at the various stages of countdown. (These details are described in the Experimental Section.)

The clock is controlled by the PRTB software through subroutines SB1, SB2, and SB10. SB1 is the *initialization* subroutine. Through it the programmer specifies the clock frequency, within the limits outlined above. (SB1 is also used to specify the symbol assigned to the variable which will take on the values of the digital data acquired in SB3 or SB7.)

SB2 is called when the computer program decides that it is time to *enable* the clock. That is, an ENCODE bit is set to a "1" state on the data acquisition channel. This bit can be connected externally with a patchcord to the ENABLE terminal on the clock module (see Fig. 2-A). Alternatively, the ENCODE bit may be brought to some external logic, the output of which will set the ENABLE clock input when other external events have occurred—like the start of the experiment. One possible external logic configuration is shown in Figure 2-B, where the ENCODE bit conditions one input of an AND gate. The other gate input is conditioned "TRUE" when the experiment has been initiated. The AND gate output will then enable the clock. At that point the countdown logic will be enabled and programmed clock pulses will begin to appear at the available outputs.

SB10 is called to disable the clock when the time base is no longer required. It simply "clears" the ENCODE bit on the data acquisition channel which has been used externally to enable the clock.

Another type of communication required between the experiment and computer is associated with *syn-*

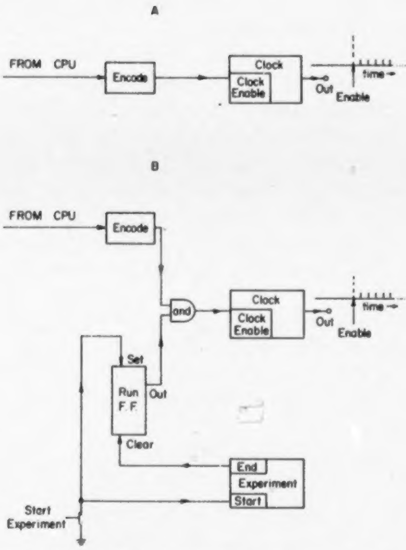

A
FROM CPU
Encode
Clock
Clock Enable
Out
time
Enable
B
FROM CPU
Encode
and
Clock
Clock Enable
Out
time
Enable
Set
Run F.F.
Out
Clear
End
Experiment
Start
Start Experiment

Figure 2. Synchronization of clock output with computer and experimental events. A. Direct computer control of clock enable input. B. Clock Enable controlled by AND gate output. Clock enabled when both computer command *and* experimental start are seen at AND gate inputs. The "RUN" flip-flop output follows the status of the experiment; it is set "true" when the experiment starts and "false" when the experiment ends. Thus, the Clock is disabled when the experiment ends.

chronization between computer operations and external events. For example, if the computer has completed all preliminary program execution required before being able to accept experimental data, the user may choose to receive an output from the computer which may not only tell the experiment to start, but may also initiate external events associated with the experimental system. One subroutine available for this kind of operation is SB5. SB5 causes the output of an ENCODE bit on a selected I/O channel to change state (go to the "TRUE" level) when SB5 is called. Then the computer waits within SB5 for an external event to occur which will cause a FLAG bit on the same I/O channel to change to a TRUE state. The subroutine detects this event and then allows the next sequential program statement to be executed. This next statement might conceivably be a call to SB2 which initiates data acquisition.

A simpler, but less precise, means of communication with the computer for the purpose of synchronization is provided by SB4. When this subroutine is entered the computer simply waits until the operator flips a toggle switch corresponding to bit 0 on the computer console switch register. The computer then exits SB4 and executes the next program statement—which might be to call SB2 and start data acquisition.

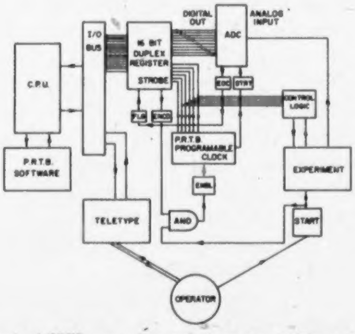

DIGITAL OUT
ANALOG INPUT
I/O BUS
16 BIT DUPLEX REGISTER
STROBE
ADC
C.P.U.
EOC
STRT
CONTROL LOGIC
FLG
ENCD
P.R.T.B. SOFTWARE
P.R.T.B. PROGRAMABLE CLOCK
EXPERIMENT
ENBL
TELETYPE
AND
START
OPERATOR

Figure 3. Typical PRTB experimental data acquisition set-up. Previously undefined symbols include: I/O Bus, Input/Output hardware of the computer, one channel of which is used for data acquisition system; 16-bit duplex register, A Hewlett-Packard #02116-6195 16-bit duplex interface buffer register card, also providing output ENCODE command and input FLAG signal.

Finally, a subroutine (SB6) is available which allows general-purpose timing functions. This routine simply waits for the FLAG bit to be set on the data acquisition channel, clears the FLAG, exits the subroutine, and the next sequential program statement is executed. The purpose of this subroutine is synchronization of program segments with the external time base. (Note that SB5 can be used, also, for general-purpose timing functions if the data acquisition channel is needed for other purposes. The user must simply connect the clock output to the FLAG input of the alternative I/O channel.)

Experimental

Computer Instrumentation

The digital computer system in this work was a Hewlett-Packard 2115A, equipped with 8K core memory, high speed paper tape input, ASR/33 Teletype, a 16-bit 20 usec DAC, and an interfaced Tektronix Model 601 oscilloscopic display. In addition, the computer has an interfaced data acquisition system and a general-purpose experimental interface capability. (These are described in detail below.) The complete computer system is mounted in a cabinet which has roll-around capability. The software used for laboratory on-line operation (PRTB) utilized the H.P. BASIC compiler, H.P. #201112A. This software is made available from the computer manufacturer. The program listings of the *additions* to the H.P. BASIC compiler made here to implement PRTB are available from the authors upon request.

Data Acquisition and General-Purpose Interface Hardware

A schematic diagram of the data acquisition system operated in conjunction with PRTB is shown in Figure 3. The analog-to-digital converter (ADC) used in the data acquisition system was manufactured by Digital Equipment Corporation (DEC), Maynard, Mass., (#C-002, 10-bit, 33 usec conversion time, 0 to −10.23 V input range).

The programmable clock is constructed from a 10 MHz

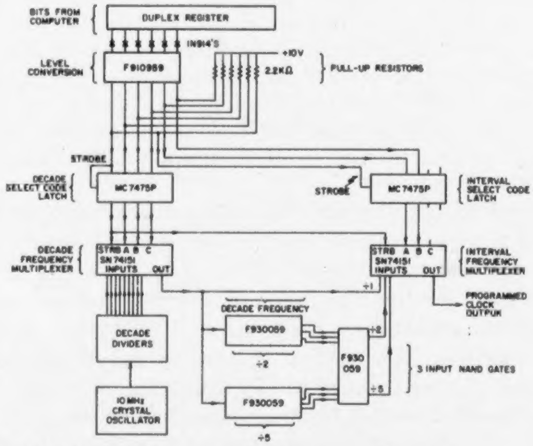

BITS FROM COMPUTER
DUPLEX REGISTER
IN914'S
LEVEL CONVERSION
F910959
+10V
2.2KΩ
PULL-UP RESISTORS
STROBE
DECADE SELECT CODE LATCH
MC7475P
STROBE
MC7475P
INTERVAL SELECT CODE LATCH
DECADE FREQUENCY MULTIPLEXER
STRB A B C
SN74151
INPUTS OUT
STRB A B C
SN74151
INPUTS OUT
INTERVAL FREQUENCY MULTIPLEXER
÷1
PROGRAMMED CLOCK OUTPUT
DECADE FREQUENCY
F930059
÷2
DECADE DIVIDERS
F930 059
3 INPUT NAND GATES
10 MHz CRYSTAL OSCILLATOR
F930059
÷5

Figure 4. Schematic diagram for PRTB programmable clock.

Device No.	Manufacturer	Description
F910959	Fairchild	Hex inverter-six Level Converters, High Level to +4 to +20 V
MC7475P	Motorola	Quad latch—4 bistable latches
SN74151	Texas Instrument	Data Selector/Multiplexer
F930059	Fairchild	4-bit shift register
F900359	Fairchild	Triple 3-input Nand Gate

crystal-controlled oscillator scaled down to useable frequency ranges with MSI (medium-scale integrated circuit) programmable countdown logic. A schematic diagram of this module is given in Figure 4, with a complete list of hardware components. Programmed clock pulses are available from 100 KHz to 0.002 Hz. Each decade frequency can be divided by 2 or 5. The specific frequency is selected by a 5-bit output word which is decoded in the clock module by 2 Texas Instrument SN74151 digital multiplexers. In addition to the programmed output, each decade output from 1 MHz to 0.001 Hz is available externally by patchboard connection.

Also available on the data acquisition panel were an ENCODE output bit which could be set and cleared by the computer, and a FLAG input terminal to allow external setting of the FLAG bit on the computer Input/Output (I/O) channel used for data acquisition. The end-of-conversion (EOC) flip-flop of the ADC was normally connected to the FLAG input; however, it was possible to connect any appropriate externally generated signal to the FLAG input. Six bits of digital information could be transferred to or from the computer thru the data acquisition panel using patchboard connections on the panel. Other generally useful functions available on the data acquisition panel included patchboard connection to various logical devices such as 16 AND, OR, and NOT gates, 4 flip-flops, 2 one-shots, 4 analog switches, 4 relay drivers, a Schmitt trigger, a track-and-hold amplifier, 6 indicator lamps, and 2 push button switches with Schmitt trigger outputs. (See Ref. (*6*) for a general discussion of characteristics of control logic nodules.) DEC R-series logic (*7*) Flip-Chip cards and power supplies were used for logic functions. For interface design which required more sophistication or more logic elements than could be obtained on the data acquisition panel, a DEC patchboard R-series Logic Lab (*7*) was available. Most experiments did not require interface hardware beyond that available on the data acquisition panel. All input and output to the computer were buffered with level conversion devices on the H.P. 2115A system so that all external connections are compatible with DEC R-series positive logic (*7*). DEC W601 and W510 level conversion cards were used.

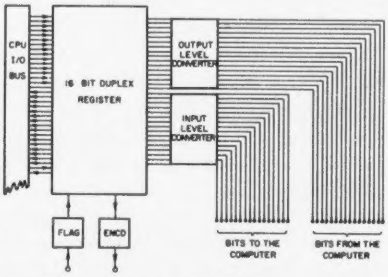


Figure 5. Digital I/O interface module for PRTB.

An interfaced digital I/O module was also available for general-purpose digital communication between computer and experiment. The characteristics of this module are illustrated in Figure 5. This digital I/O Module was connected to a different I/O channel than the data acquisition panel. Thus, a separate independent ENCODE output and FLAG input were available thru this module. The use of the digital I/O capability has been described above.

Results and Discussion

Evaluation of the PRTB System

For the purpose of illustrating the capabilities of the PRTB software for this publication, the laboratory computer systems and software were tested with two synthetically generated experimental outputs which simulated typical experimental data (*4*, *8*). The first of these involved a transient experiment with a rapid exponential voltage decay output. The second illustration involved application of ensemble averaging to the analysis of an experimental output which provided a repetitive voltage waveform with superimposed large amplitude random noise.

```
READY
LIST
10 REM THIS IS A PROGRAM THAT WILL PRINT OUT THE DECIMAL NUMBER
20 REM EQUIVALENT OF AN ANALOG EXPONENTIAL DECAY CURVE.
30 DIM X(250)
40 LET F=100
50 LET T=250
60 CALL (1,X(I),F)
70 CALL (7,T)
80 FOR I=1  TO  250
90 PRINT X(I);
100 NEXT I
110 END
```

Figure 6. Program for data acquisition from exponential decay function.

Table 2. Results from Execution of Program in Figure 6

Print-out of exponential decay data from program of Figure 6

```
  960  956  951  947  943  939  935  931  927  923  919  916
  912  908  904  900  896  892  888  885  881  877  873  870
  866  862  859  855  852  848  844  841  837  834  830  827
  823  820  816  813  809  806  803  799  796  793  789  786
  783  779  776  773  770  766  763  760  757  754  751  747
  744  741  738  735  732  729  726  723  720  717  714  711
  708  705  702  699  696  693  690  687  685  682  679  676
  673  670  668  665  662  659  657  654  651  649  646  643
  641  638  635  632  630  627  625  622  619  617  614  612
  609  607  604  602  599  597  594  592  589  587  584  582
  580  577  575  572  570  568  565  563  561  558  556  554
  552  549  547  545  543  540  538  536  534  532  529  527
  525  523  521  518  516  514  512  510  508  506  504  502
  500  498  495  493  492  490  488  486  484  482  480  478
  476  474  472  470  468  466  464  462  460  458  457  455
  453  451  449  447  445  444  442  440  438  436  435  433
  431  429  428  426  424  422  421  419  417  416  414  412
  411  409  407  406  404  402  401  399  397  396  394  393
  391  390  388  386  385  383  381  380  378  377  375  374
  372  371  369  368  366  365  363  362  361  359  368  356
  355  353  352  350  349  348  346  345  343  342
READY
```

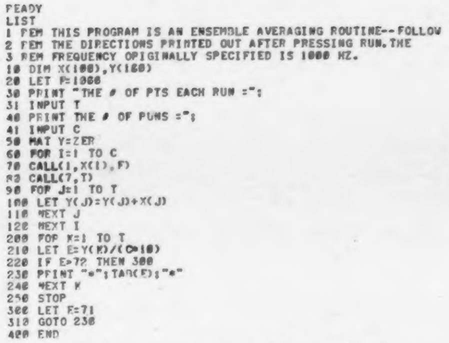
The exponential decay waveform had a time constant the order of 0.1 sec. Data were taken at a constant rate for a fixed length of time, and no processing of the data was carried out. The data were acquired, stored, and the digitized data printed out at the completion of the experiment. The program used is given in Figure 6, and a typical output is given in Table 2.

The waveform for the ensemble averaging experiment had a fundamental frequency the order of 10 Hz, with superimposed large amplitude random noise. The experiment was repeatable and could be triggered by computer output. The programming objectives included repetitive initiation of the experiment, synchronization of data acquisition with experimental output, and repetitive coherent summation of the digitized waveforms to accomplish ensemble averaging. Finally, when the experiment was completed the averaged data were normalized and plotted on the teletype terminal. The computer program used is given in Figure 7. Figure 8 shows the original output waveform and the results of ensemble averaging.

Observations

It can be seen from the above examples that actual implementation of the PRTB system is relatively straightforward. Moreover, for all its simplicity, considerable experimental measurement capability exists in the system.

The requirements for implementing the PRTB system in the undergraduate laboratory include the following



```
FEADY
LIST
1 FEM THIS PROGRAM IS AN ENSEMBLE AVERAGING ROUTINE--FOLLOW
2 FEM THE DIRECTIONS PRINTED OUT AFTER PRESSING RUN.THE
3 REM FREQUENCY OPIGINALLY SPECIFIED IS 1000 HZ.
10 DIM X(100),Y(100)
20 LET F=1000
30 PRINT "THE # OF PTS EACH RUN =";
31 INPUT T
40 PRINT THE # OF PUNS =";
41 INPUT C
50 MAT Y=ZER
60 FOR I=1 TO C
70 CALL(1,X(1),F)
80 CALL(7,T)
90 FOP J=1 TO T
100 LET Y(J)=Y(J)+X(J)
110 NEXT J
120 NEXT I
200 FOP K=1 TO T
210 LET E=Y(K)/(C*10)
220 IF E>72 THEN 300
230 PFINT "*";TAB(E);"*"
240 NEXT K
250 STOP
300 LET E=71
310 GOTO 230
400 END
```

Figure 7. Program for ensemble averaging experiment.

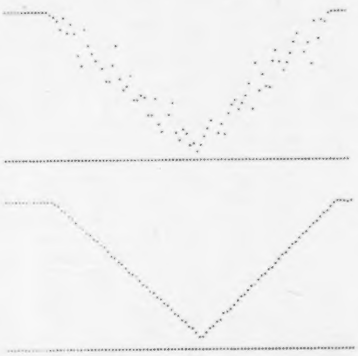


Figure 8. Data from ensemble averaging experiment, plotted on teletype. Above, Original data (single cycle). Below, Data after 100 averaging cycles.

1. an introduction to the BASIC language for computational purposes. This requires 1 to 2 hr of lecture time. It also requires that students have access to the laboratory computer (or other computer facilities providing BASIC capability) during the first few weeks of the semester to handle homework and laboratory computations.
2. an introduction to the fundamental concepts of on-line computer operation. This requires about 6 hr of lecture. These introduce the student to the whole field of on-line computer applications and the technological details with which they must be familiar to implement this approach in the laboratory. These details include an introduction to the fundamentals of digital logic, timing, and synchronization. This involves the discussion of simple gates (AND, OR, NOT), the flip-flop, the one-shot, and the analog switch. The functional characteristics of these devices, as well as of analog-to-digital and digital-to-analog converters and voltage amplifiers, for interface design are defined. No attempt is made to provide a rigorous understanding of the electronic principles or detailed circuitry for these devices. The emphasis is on the student being able to recognize and use the fundamental characteristics of these devices for interface design.
3. finally, the actual laboratory assignments are carried out. These assignments are designed to introduce the student to the analytical technique and methodology, just as would be done in the absence of the laboratory computer instrumentation. In addition, the student is expected to utilize the on-line computer as a data acquisition and data processing tool.

The final question to be considered here is what specific benefits are derived by the student from the use of the laboratory computer in on-line experimentation. First of all, the student is obviously exposed to state-of-the-art technology in laboratory experimentation. Secondly, the student is encouraged to use more rigorous data processing approaches than previously feasible for laboratory assignments. Thirdly, the student becomes keenly aware of the factors which limit and define experimental accuracy and precision. Fourthly, the student is able to make a first hand comparison of the effectiveness of conventional and computerized laboratory methodology. And, finally, the student's interest in the science of quantitative chemical experimentation is strongly stimulated.

Future Work

Several different experiments involving on-line computer studies are currently being developed which are appropriate for incorporation into the undergraduate analytical laboratory. These include kinetic methods of analysis, amperometric titrations, coulometric titrations, potentiometric titrations, spectrophotometric analysis, fast-sweep polarographic analysis, and thermal analysis methods, in addition to synthetic and chromatographic experiments already developed (*8*, *9*). Some of these experiments will be included in the introductory course and some in an advanced analytical course (Senior level at Purdue).

It is also desired to further improve the PRTB software so that the experimental subroutine calls can be replaced by conversational macro-instructions.

Acknowledgment

This work supported by a grant from the National Science Foundation Office of Computing Activities, Grant No. GJ-428. The authors wish to thank Dr. David O. Jones for technical advice, and Professors L. B. Rogers, H. L. Pardue, and F. E. Lytle for their respective contributions to the suggestion, encouragement, criticism, and technical support of this program.

Literature Cited

(1) Frazer, J. W., *Anal. Chem.*, **40**, 26A (1968).
(2) Kuzel, N. R., Roudebush, H. E., and Stevenson, C. E., *J. Pharm. Sci.*, **58**, 381 (1969).
(3) Perone, S. P., *J. Chrom. Sci.*, **7**, 714 (1969).
(4) Perone, S. P., J. Chem. Educ., **47**, 105 (1970).
(5) Kemeny, J. G., and Kurtz, T. E., "BASIC Programming," John Wiley & Sons, N.Y., 1967.
(6) Malmstadt, H. V., and Enke, C. G., "Digtal Electronics for Scientists," W. A. Benjamin, Inc., N.Y., 1969.
(7) "Digital Logic Handbook," Digital Equipment Corporation, Maynard Mass., 1968.
(8) Jones, D. O., submitted for publication.
(9) Perone, S. P., and Eagleston, J. F., in press, J. Chem. Educ.

◆ ◆ ◆

[Reprinted from Journal of Chemical Education, Vol. 48, Page 438, July, 1971]
Copyright 1971, by Division of Chemical Education, American Chemical Society, and reprinted by permission of the copyright owner

S. P. Perone
and **J. F. Eagleston**
Purdue University
Lafayette, Indiana 47907

On-Line Digital Computer Applications in Gas Chromatography

An undergraduate analytical experiment

An experiment in gas chromatography provides an ideal background for an introduction to on-line computer instrumentation. The experimental concepts are relatively simple and the apparatus is readily available and reliable; a wide variety of data processing problems arise; and data handling can be a serious problem.

In this publication are presented some descriptive background material and the experimental directions provided for the students to whom the experiment is assigned. It is assumed that the student is familiar with the Purdue Real-Time BASIC (PRTB) laboratory computer system, which has been described previously (*1*).

Background

Gas Chromatography (G.C.)

G.C. is an analytical technique to achieve quantitative separation of mixtures. It is applicable to any samples which can be vaporized and introduced to the separation process in a gaseous state.

A schematic diagram of the chromatographic instrumentation is shown in Figure 1. More detailed discussions of G.C. theory, instrumentation, and procedures are given in reference (*2*).

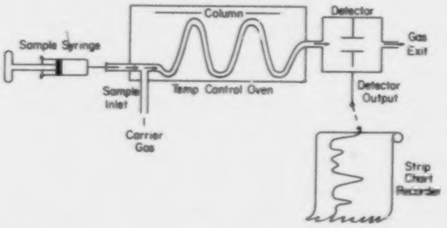


Figure 1. Block diagram of G.C. instrumentation.

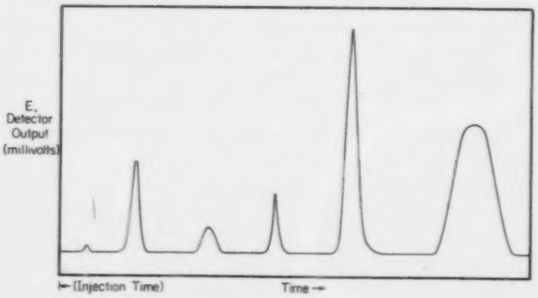


Figure 2. Typical chromatogram for well-separated components.

Quantitative Measurements in G.C. Figure 2 illustrates a typical chromatographic output—or "chromatogram"—for a multi-component mixture where each component has been well separated. Quantitative data are obtained by determining the area for each peak, A_i. If the detector *response factor* is identical for each component, the % composition for the *i*th component is given by $(A_i/\Sigma A_i) \times 100\%$.

The area of a peak can be determined *manually* from the chart paper record of the output by triangulation, counting squares, or planimeter measurement.

Qualitative Measurements. The precise elution time corresponding to the peak maximum (*retention time*) can be used for identification of each component in the sample. This feature depends on having retention time data available for pure standards run under identical conditions of column material and size, length, temperature, and flow rate.

Computer Processing of Chromatographic Data

Assuming that chromatographic data can be tabulated and entered into the computer's memory, it becomes possible to use a computer program to process the data. Not only can the simple manual methods described above be replaced, but processing problems which are too complex for manual methods can be considered and handled. We will defer the discussion until later regarding how chromatographic data are to be transferred to computer memory. First, let us consider how the data can be processed once it is acquired by the computer.

Peak Integration. The peak area is directly proportional to the *integral* of the peak waveform. Figure 3 and eqn. (1) show this relationship.

$$A \alpha Q = \int_{t_1}^{t_2} (y - b) dt \qquad (1)$$

Here, Q is the value of the integral, t is time, y is the value of the G.C. output as a function of time, and

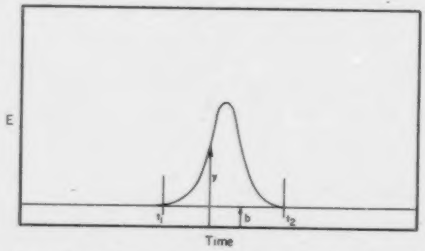


Figure 3. Integration limits and variables for peak area determination.

b is the base-line value (which may or may not be time-dependent). Computer integration of data is a very simple operation, but one must consider first what information is available in memory, and how it can be used.

The G.C. data could be tabulated, for example, such that output values are measured at 1-sec intervals along the entire length of the curve and saved in memory. Assuming a total of 200 data points, the result would be an array where $Y(1)$ corresponds to the first datum in the array and the first time interval ($t = 1$ sec); $Y(100)$ corresponds to the 100th datum in the array and a time of 100 sec; and so on. This translation is shown in Figure 4. The computer program can access any element in the array (any data point) by specification of the proper subscript. Moreover, according to the storage scheme of Figure 4, the subscript also corresponds to the time of the data point.

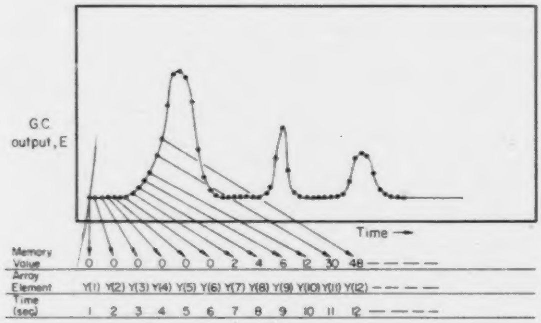


Figure 4. Correspondence between experimental output and tabulated array in computer memory.

To generate a program to integrate peak areas, use the fact that an integral is defined by a summation of $y(t)\Delta t$ as Δt approaches zero

$$Q \simeq \sum_{i=1}^{n} y_i \Delta t_i \tag{2}$$

For constant Δt

$$Q \simeq \Delta t \sum_{i=1}^{n} y_i \tag{3}$$

The program might have the following features: assuming a zero base-line, examine each data point consecutively; when data are consistently above some arbitrary *threshold*, assume the data are on the peak; add consecutive data points together until the data are consistently below the threshold again; the sum generated, when multiplied by the constant, Δt, is equivalent to the peak integral. (Note that a constant data sampling interval must be assumed. Also note that if only relative values of integrals are required—as is usually the case—it is not necessary to multiply the sum by Δt.) The *thresholding* algorithm described above is illustrated in the programming flow chart of Figure 5.

Other Processing Objectives

In addition to the integration of individual peaks, several other processing functions may be required for gas chromatographic data. Some of these will be outlined briefly here. Fewer details regarding the processing algorithms are presented, as these will be left to the student's discretion.

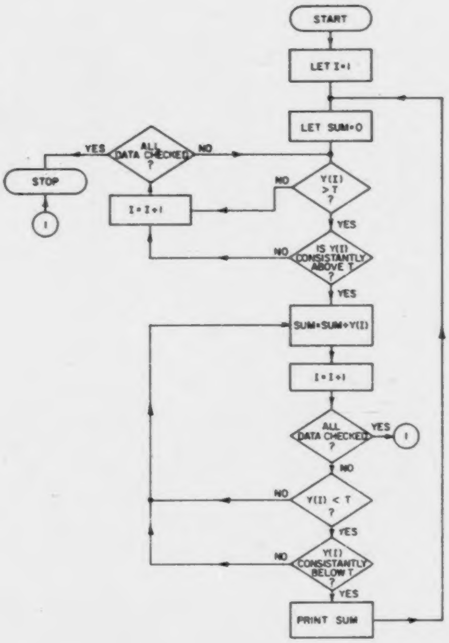


Figure 5. Flow chart for program to integrate area under each peak in a chromatogram and print each result. Based on simple thresholding algorithm.

Precise Peak Location and Identification. The establishment of the time associated with the peak maximum (Retention Time) is important for identifying the component giving rise to the peak. The rigor with which one attempts to measure the retention time depends on several factors: (1) the accuracy with which the time base is known—i.e., the accuracy with which "zero" time has been established and the accuracy of the data acquisition timing; (2) the reproducibility of the timing characteristics and the experimental data; and (3) the accuracy required for distinguishing qualitatively between possible components of the mixture.

Obviously, if the inherent accuracy and/or reproducibility of the experiment are poor, then it is foolish to develop a rigorously accurate peak location program. Moreover, even if experimental accuracy and reproducibility allow rigorous programming, the required accuracy may be so undemanding that rigorous peak location processing is not warranted. A possible approach to establishing the retention time for each peak in this case would be to assume symmetrical peaks and compute the mid-point between the points where the data went above and below the threshold.

Should rigorous determination of retention times be both required and experimentally possible, alternative processing algorithms should be considered. For example, the peak data could be differentiated and the point at which the derivative goes through zero equated to the peak maximum. This is not a straightforward matter however, because the data in the computer's memory are discontinuous. Thus, mathematical differentiation can be achieved either by taking finite differences, or by fitting a quadratic or higher order equation to the peak maximum and differentiating the equation. If the data are not completely *smooth*, the second alternative is preferred. (These algorithms and curve-fitting procedures are discussed in detail in ref. (*3–7*).)

Non-Uniform Detector Response. To determine percent composition from peak areas, one must either be able to assume uniform detector response to all components—as described earlier—or else the individual response factors must be known. Assuming standard runs with each of the possible mixture components can be obtained, the response factors can be tabulated. Mathematical implementation of these factors to calculate correct percent compositions from peak areas should be obvious. The program development is left to the student.

Resolution of Overlapping Peaks. Frequently gas chromatographic separations are incomplete because the optimum experimental conditions are unknown or unattainable. In such cases some chromatographic peaks may overlap as shown in Figure 6.

Algorithms for resolving overlapping G.C. peaks have been discussed extensively in the literature (*8–10*). Many of these utilize the first- and second-derivatives of the G.C. data to recognize peak characteristics. These relationships are summarized in Figure 6 of reference (*8*). It is left to the student to develop algorithms for G.C. data processing based on these characteristics. Perusal of the literature references given is strongly suggested.

Non-Zero Base-Lines. Up to this point we have assumed a base-line or background level which is zero or constant. However, it is not unusual to observe large base-line drifts in chromatographic experiments, as shown in Figure 7. Problems of this nature are particularly prevalent when temperature programming is employed, as "column-bleed" may occur at elevated temperatures and contribute to the background level.

The data processing algorithm to handle background drift could be similar to that designed to handle shoulders on large peaks. Estimating the base-line for a given peak could be as simple as projecting a linear segment tangent to adjacent minima in the curves, or as complex as projecting a curved segment fit to the shape of the minima on either side of a peak. The experimenter must decide on how necessary the exotic processing program might be. This is generally based on the frequency with which difficult base-line problems are encountered in the systems studied. Generally, it is more practical to put effort into improving the experimental systems to avoid difficult data processing situations, and utilize relatively straightforward processing algorithms.

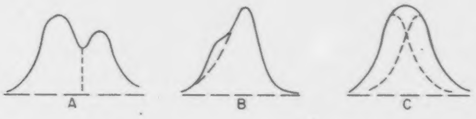


Figure 6. Examples of unresolved G.C. peaks. (A) Two identifiable maxima, (B) one shoulder, one maximum, (C) two peaks fused to give one maximum.

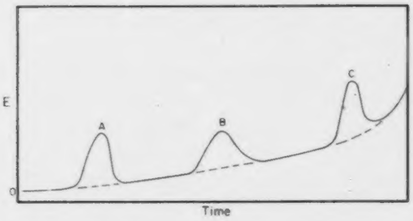


Figure 7. Example of changing chromatographic base line.

Accurate Area Determinations. The thresholding algorithm suggested above as a method for peak area measurements is obviously inaccurate. This is particularly so when applied to very small or very broad peaks. For large and/or sharp peaks the errors can usually be neglected.

In cases where accurate peak area determinations are required, more rigorous processing algorithms must be used. More critical estimates of the points where the data begin to deviate from the background, and subsequently return to the background level, are required. The best approach probably is to look at the 1st and 2nd derivatives, equating the start of the peak to the point where the 2nd derivative changes from a consistent low or zero value to a definite positive value. The end of the peak should see the 2nd derivative approaching a consistent low or zero value from the positive direction.

Noisy Data. Experimental data are rarely free from random fluctuations and background electronic noise. Because of this fact, the straightforward application of any processing algorithm which assumes "smooth" data curves may lead to serious misinterpretation of the data or complete failure. The magnitude of the problem is inversely related to the Signal-to-Noise Ratio (S/N) of the data. If S/N never gets below 100:1, the data processing program that assumes "smooth" data [illegible] probably work. When S/N falls somewhat below this value, one can no longer ignore the noise in the data processing program.

The first step in solving the problem of processing noisy data is to attempt to isolate the instrumental source of the noise and minimize it. If the experimenter has implemented all reasonable means to minimize instrumental noise contributions and the data still contain significant noise background, the only choice is to employ mathematical smoothing techniques.

In this approach a program is written which first processes the raw data to obtain a new smooth curve which hopefully fits the fundamental data. There are various mathematical approaches available. The student is referred to the article by Savitzky and Golay (*7*) which provides a detailed discussion of smoothing methods and provides details for a least-squares technique to obtain smooth 0, 1st, and higher derivatives from raw noisy data. Once the smoothed data are obtained the data processing programs for extraction of chemical information may be applied.

Qualitative Identification. Computer identification of mixture components can be accomplished by establishing a table of standard retention times. A program to compare experimentally observed retention times with the tabular values can then achieve qualitative identification. There are two significant points to be kept in mind for this: (1) The retention time standard data must be obtained under identical experimental conditions to those for the unknown data; and (2) the comparison of observed retention times with the tabulated standards must allow for some experimental uncertainty in the measurements. A common approach is to assign some identification "window" to each standard retention time, such that any experimentally observed retention time that falls within that window is associated with that standard. The size of the window selected is a function of experimental reproducibility, the magnitude of the retention time, and the anticipated composition of mixtures. For example, a window of ±10 sec might be assigned to a retention time at 100 sec when the next closest expected component has a retention time of 200 sec. However, if the next component has a retention time of 115 sec, it is necessary to establish a narrower I.D. window (e.g., ±5 sec) if experimental reproducibility allows it.

On-Line Computer Programming and Instrumentation

To this point the G.C. data processing considerations have been presented independent of the manner in which the computer is connected to the experimental system. Now, we will consider the programming and instrumentation necessary for *on-line* computer operation, where the computer monitors the experiment and controls data acquisition.

A block diagram of the instrumentation and interfacing required for on-line execution of the G.C. experiment is shown in Figure 8. All of the interface components are provided in the laboratory. It is up to the student to make the appropriate connections, to select the proper conditions for operation, and to verify that the interface is working properly.

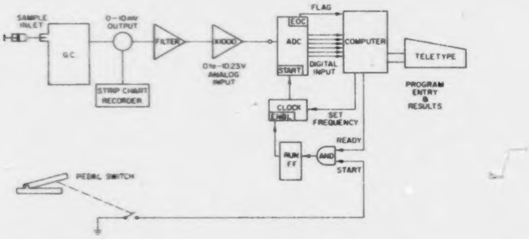


Figure 8. Block diagram of instrumentation and interfacing for on-line computer system for G.C. experiment.

Description of Interface Components

The basic interface and data acquisition characteristics for the system shown in Figure 8 are described in detail in reference (*1*). The student should be familiar with this description. For this experiment, special interface components include a voltage amplifier and filter and a foot pedal switch. The amplifier-filter electronics have been prepared on a printed circuit card which can be plugged into the general-purpose interfacing panel (GIP) in the laboratory. The input terminal should receive the direct G.C. output; and the amplifier-filter output should be connected to the analog-to-digital converter (ADC) analog input terminal.

The foot switch is also provided in the laboratory. It is used to synchronize the sample injection with the start of the automatic data acquisition process. The electrical input and output of the foot switch can be accessed on the GIP. The foot switch output can be used to condition the AND gate input to enable the CLOCK at the start of the experiment. (The other AND gate input must have been set by the computer when it was ready to accept experimental data (*1*).)

The data acquisition rate is determined by the CLOCK frequency. That frequency can be set by the experimenter's program as described in reference (*1*).

On-Line Programming Features

The experimenter's program must include the following considerations: (1) selection of data acquisition rate, (2) synchronization with the start of the experiment, (3) data storage, and (4) "real-time" and "non-real-time" data processing.

Selection of the data acquisition rate must be based on some knowledge of the desired data density. A rate which provides about 20 or 30 data points on a peak is adequate.

Synchronization can be accomplished by calling one of the experimental synchronization subroutines defined in reference (*1*) describing the PRTB software.

Data storage can be a serious problem. The computer has finite memory space available. Moreover, in the BASIC configuration employed here, data storage is available only in 255-word blocks. Thus, if one were to store all data points from an experiment, the total number could not exceed 255 unless the experimenter made provision to link blocks during the data acquisition process. However, a maximum of only 500 to 1000 data points could be stored in the computer systems used here—assuming minimal program space. Thus, for a typical G.C. experiment that runs several minutes, with a required data acquisition rate of 10 pts/sec, there would not be enough storage space available.

At this point, the experimenter must consider "real-time" data processing—i.e., processing the data *as it is being acquired*. One advantage of this type of processing is that not every data point need be stored. For example, the simplest approach might involve looking at data as they come in, deciding whether or not the data are on a peak, saving only those data corresponding to peaks, and saving the corresponding times for start and finish of each peak. Using this approach, a chromatogram containing 10 peaks might require only 200 or 300 storage spaces in memory. When data acquisition was completed, the program could fetch the stored data and process as desired in "non-real-time."

Many other "real-time" data processing functions might be considered. For example, the data might be integrated as it is taken. Thus, at the end of each peak the peak integral would be available. The limitation to "real-time" programming is the amount of time available between data points and the time required for program execution between data points. The PRTB system will provide an error diagnostic print-out whenever too much program execution is attempted between data points. The experimenter can use trial and error, but a useful guideline to follow is that each BASIC program statement requires about 10 msec execution time.

An example of a program using the PRTB software (*1*) for G.C. data acquisition and simple real-time integration of each peak (using a thresholding algorithm) is given in Figure 9. The program corresponds to the flow chart of Figure 5. Figure 10 shows the initial dialogue between computer and operator, as well as the results print out for a typical chromatographic experiment with a 3-component sample.

Laboratory Assignments

The following assignments are listed in the order of progressively increasing difficulty. Each student should complete assignments (a) thru (c). For advanced work complete any of the additional assignments. Two to four 4-hr lab session equivalents are required, depending on number of alternative studies pursued. Programs must be developed outside of normal lab time.

Assignments

I. *Assuming well-separated peaks and zero base-lines for assignments (a) to (e)*
  (a) Enter synthetic G.C. data from paper tapes supplied by teaching assistant in the lab. Process the data with a simple thresholding algorithm to determine peak areas and percent composition of each component. Determine retention times using t_r = mid point of threshold segment.
  (b) Interface G.C. to the computer and execute the data acquisition program segment, simply printing out the acquired data block.
  (c) Take complete data block on-line and process as in (a).
  (d) Same as (c), except provide for correction for detector response factors.
  (e) Take G.C. data on-line and use "real-time" data processing to integrate each peak as it is observed and provide same processed data as in (a), except don't save any data points permanently.
II. *Without assuming well-separated peaks or zero base-lines*
  (f) Use an appropriate on-line processing approach to handle overlapping peaks.
  (g) Use an appropriate on-line processing approach for handling non-zero and non-constant base-line problems.
  (h) Use a mathematical smoothing approach to satisfactorily handle noisy data.
  (i) Use an appropriate processing approach for obtaining very *accurate* peak area and retention time measurements.

Required Materials and Equipment

A. Chemicals.
  1. For 3 well-defined peaks, zero base-line: mixture of *n*-heptane, *n*-octane, and *n*-nonane.[1]
  2. For base-line drift w/o temp programming[2]

fair results	.5 ml pentane	and 10 ml MeOH	time 1.5 min
good results	.5 ml hexane		time 2 min
v. good results	.5 ml heptane		time 6 min

  3. Fused peaks

fair —mixture of pentane, hexane, and heptane[1]
good—mixture of MeOH, pentane, hexane[2]

B. Gas Chromatograph-Aerograph 202 with 1 10-ft 20% SF-96 column and 1 6-ft 20% Carbowax 30 m. column and 1 Esterline Angus (Speed Servo) recorder 1 10-μl syringe.
C. DEC Logic Lab (Digital Equipment Corp., Maynard, Mass.).
D. 1 filter and amplifier card for General Purpose Interface Panel.
E. Foot switch.
F. H-P 2115A Computer w/8K core memory, teletype, high-speed paper tape reader, General Purpose Interface Panel, and programmable clock.
G. PRTB Compiler (*1*).

Literature Cited

(1) Perone, S. P., Eagleston, J. F., J. Chem. Educ., **48**, 317 (1971).
(2) Schupp, O. E., III, in "Technique of Organic Chemistry," Vol. XIII, (*Editors:* Perry, E. S., and Weissberger, A.), Interscience Publishers, N. Y., **1968**.
(3) Bevington, P. R., "Data Reduction and Error Analysis for the Physical Sciences," McGraw-Hill, N.Y., **1969**.
(4) Groves, W. E., "Brief Numerical Methods," Prentice-Hall, Inc., Englewood Cliffs, N. J., **1966**.
(5) Mandel, J., "The Statistical Analysis of Experimental Data," Interscience, N. Y., **1964**.
(6) Ralston, A., "A First Course in Numerical Analysis," McGraw-Hill, N. Y., **1964**.
(7) Savitzky, A., Golay, M. J. E., *Anal. Chem.*, **36**, 1627 (1964).
(8) McCullough, R. D., *J. Gas Chrom.*, **5**, 635 (1967).
(9) Baumann, F., Brown, A. C., Mitchell, M. B., *J. Chrom. Sci.*, **8**, 20 (1970).
(10) Hancock, H. A., Jr., Dahm, L. A., Muldoon, J. F., *J. Chrom. Sci.*, **8**, 57 (1970).

```
READY
LIST
10 PRINT "THE THRESHOLD (IN MILLIVOLTS) WILL BE=";
11 INPUT A
20 PRINT "TOTAL # OF DATA POINTS WILL BE=";
21 INPUT B
30 PRINT "THE DATA ACQUISTION RATE WILL BE=";
31 INPUT C
40 LET D=E=F=S=G=0
50 DIM X(1),Y(10)
60 CALL(1,X(1),C)
70 CALL(2)
80 IF D=B THEN 410
90 CALL(3)
100 LET D=D+1
110 IF X(1)>A THEN 130
120 GOTO 80
130 LET E=E+1
140 IF E=3 THEN 210
150 CALL(3)
160 LET D=D+1
170 IF D=B THEN 410
180 IF X(1)>=A THEN 130
190 LET E=0
200 GOTO 80
210 LET F=F+1
220 LET E=0
230 LET S=X(1)+S
240 CALL(3)
250 LET D=D+1
260 IF D=B THEN 390
270 IF X(1)<=A THEN 290
280 GOTO 230
290 LET E=E+1
300 IF E=3 THEN 360
310 CALL(3)
320 LET D=D+1
330 IF D=B THEN 390
340 IF X(1)<=A THEN 290
350 GOTO 220
360 LET Y(F)=S
370 LET E=S=0
380 GOTO 90
390 PRINT "PEAK #;F;"IS INCOMPLETE!"
400 LET Y(F)=S
410 CALL(10)
420 FOR I=1 TO F
430 LET G=Y(I)+G
440 NEXT I
450 PRINT "PEAK#";TAB(10);"PEAK AREA (NORMALIZED)"
460 FOR I=1 TO F
470 PRINT I;TAB(16);Y(I)/G
480 NEXT I
490 END

READY
```

Figure 9. Program for G.C. data acquisition and simple real-time integration of each peak using a threshold algorithm.

```
READY
RUN
THE THRESHOLD (IN MILLIVOLTS) WILL BE =?100
TOTAL # OF DATA POINTS WILL BE=?1000
THE DATA ACQUISTION RATE WILL BE=?10
PEAK #       PEAK AREA (NORMALIZED)
 1                     .573594
 2                     .221683
 3                     .204723

READY.
```

Figure 10. Print-out of results of chromatographic experiment.

---

[1] 6 ft Carbowax 30 m column 0.5 ml/sec flowrate-Column, detector, and injector temperatures were respectively 79°C, 200°C, 129°C. (These 3 were the same for all other samples.)

[2] 10 ft 20% SF-96 Column, 1 ml/sec flow rate.

D. O. Jones, M. D. Scamuffa,
L. S. Portnoff, and **S. P. Perone**
Purdue University
Lafayette, Indiana 47907

On-Line Digital Computer Applications to Kinetic Analysis

An undergraduate experiment

The experiment described here is one in which both kinetic methods of analysis and a computer employed in an on-line configuration are used to solve an analytical problem. It is assumed that the student is familiar with the Purdue Real-Time BASIC (PRTB) laboratory computer system, which has been described previously (*1*, *2*).

While the main purpose of the experiment is to introduce the student to reaction rate measurements (*3*) and kinetic methods of analysis (*4*), it also introduces him to the use of an enzyme as an analytical reagent (*5*). Operational amplifiers are employed to make the analog data acceptable for input into the computer, and therefore, the experiment can also illustrate circuit design.

Background

The experiment concerns itself with determining the concentration of glucose in the 10–100 ppm range by measuring its oxidation rate in the presence of the enzyme glucose oxidase. The reaction is

$$\text{RCHO} + \text{O}_2 + \text{H}_2\text{O} \underset{\text{Oxidase}}{\overset{\text{Glucose}}{\rightleftharpoons}} \text{RCOOH} + \text{H}_2\text{O}_2 \qquad (1)$$

The hydrogen peroxide produced reacts rapidly with iodide ion to produce an equivalent amount of iodine (*6*, *7*).

$$\text{H}_2\text{O}_2 + 2\text{I}^- + 2\text{H}^+ \xrightarrow{\text{Mo(VI)}} \text{I}_2 + 2\text{H}_2\text{O} \qquad (2)$$

The utilization of a molybdate catalyst (Mo(VI)) insures that reaction 2 is faster than the enzymatic reaction so that the rate of iodine production is equal to the rate of glucose consumption. The overall reaction is

$$\text{RCHO} + \text{O}_2 + 2\text{I}^- + 2\text{H}^+ \xrightarrow[\text{Mo(VI)}]{\substack{\text{Glucose}\\ \text{Oxidase}}} \text{RCOOH} + \text{H}_2\text{O} + \text{I}_2 \qquad (3)$$

The rate at which iodine is produced is given by

$$\frac{\text{d}[\text{I}_2]}{\text{d}t} = KG_t \qquad (4)$$

where G_t is the glucose concentration at time t and K is a pseudo-first-order rate constant. A rotating platinum electrode is used to detect the rate of iodine production by monitoring the electrolysis current, i_t, at the electrode. The electrolysis current is proportional to the iodine concentration at time t

$$i_t = K'[\text{I}_2]_t \qquad (5)$$

where K' is dependent upon characteristics of the cell, electrode, and polarizing voltage. The change in current from time t_1 to t_2 is given by

$$\Delta i = K'\{[\text{I}_2]_2 - [\text{I}_2]_1\} = K'(\Delta[\text{I}_2]) \qquad (6)$$

If the current interval is small enough to insure that the total change in the glucose concentration will be small, the iodine concentration will change *linearly*. Thus, $\text{d}[\text{I}_2]/\text{d}t$ in eqn. (4) can be approximated by $\Delta[\text{I}_2]/\Delta t$. Solving eqn. (4) for $\Delta[\text{I}_2]$ and substituting this value into eqn. (6) gives

$$G_t = \left(\frac{\Delta i}{KK'}\right)\left(\frac{1}{\Delta t}\right) \qquad (7)$$

Substituting $G_t = G_0 - i_t/K'$ into eqn. (7) one obtains

$$G_0 = \frac{1}{K'}\left\{\left(\frac{\Delta i}{K}\right)\frac{1}{\Delta t} + i_t\right\} \qquad (8)$$

Thus, eqn. (8) predicts that the glucose concentration is a linear function of the reciprocal of the elapsed time (Δt) required to traverse a predetermined current interval, Δi. It has previously been shown (*8*) that eqn. (8) is valid to within 1% if the measurement interval consumes less than 5% of the total glucose present when the measurement is started.

Instrumentation

The reader is directed to reference (*1*) for details concerning the computer instrumentation used in this work. The general features are computer-independent and could be implemented with any one of a variety of computers as indicated in reference (*1*).

Reaction Cell

The reaction takes place in a small cell having a total capacity of about 5 ml. The cell is immersed in a thermostated liquid in order to facilitate temperature control. [The details of cell dimensions and apparatus can be found in reference (*8*).]

Measurement Circuit

The electronic circuitry used in the experiment is depicted in Figure 1. This circuitry performs the function of controlling the potential of the monitoring electrode, measuring the electrolysis current, and making the analog data acceptable for input into the computer.

The computer will only monitor voltages between 0 and −10.23 V. Therefore, the signal from the

---

This work was supported by a grant from the National Science Foundation Office of Computing Activities, Grant No. GJ-428.

[Reprinted from Journal of Chemical Education, Vol. 49, Page 717, October, 1972.]
Copyright, 1972, by Division of Chemical Education, American Chemical Society, and reprinted by permission of the copyright owner

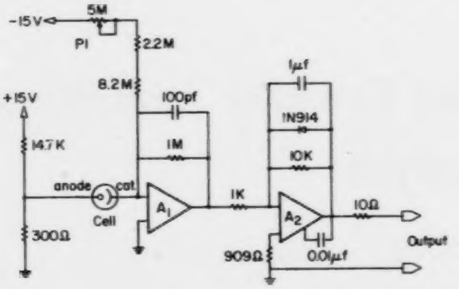


Figure 1. Control and monitoring circuitry. A_1 = Analog Devices P501A; A_2 = Fairchild 741 operational amplifier. Each amplifier decoupled by providing 0.01 mfd capacitors between +15 V and ground and −15 V and ground.

reaction cell must be converted to a voltage and amplified by a large factor.

The −15 V input to the summing point of amplifier A_1 in Figure 1 is included to bias the output of A_1 to about +1.0 V. Because the gain of A_2 is −10, the initial output bias is about −10 V. When electrolysis current increases in the cell, the output of A_1 goes negative while the output of A_2 goes positive. The diode in the feedback loop of A_2 prevents the output of A_2 from going positive of zero. Thus, the output is kept within the acceptable input range of the ADC, 0 → −10.23 V. (It should be noted that, for our computer system, −10.23 V input yields a digital value of +10,230 to the computer. Thus, during an experimental run, the digitized data appear to *decrease* with time.)

Potentiometer P1 should be adjusted at the beginning of each experiment, as the buffer and enzyme solutions may vary in residual current. However, if a series of experiments are being run where only analytical Δi values are measured, the bias voltage need not be adjusted for each run, as long as the lower threshold for Δi is about 0.1 μA above the initial bias value.

The amplification factors for the circuitry of Figure 1 provide a 10 V output swing for a 1 μA change in cell current (0.1 μA/V conversion factor). This represents the optimum condition for the analytical experiments described here and for the particular cell and electrode characteristics. The amplification factor can be lowered satisfactorily by decreasing the feedback resistance for A_2. This would be necessary for experiments to monitor total reaction times.

Computer Configuration

The schematic diagram for the overall experimental set-up, including the on-line computer, is given in Figure 2. The specific interfacing logic and connections made will depend on the particular experimental objectives. For example, if only a measurement of Δt for traversement of a specified current interval after initiation of the reaction is required, no critical synchronization features need be incorporated in the interface. However, if it is desired to monitor the entire course of the reaction, starting at t_0, the computer must be made aware of the instant when sample is injected. This can be accomplished easily within the PRTB system by utilizing manual sense switches (*1*). An uncertainty of ±0.5 sec in t_0 can be tolerated for this experiment, as reagent mixing is not instantaneous and the reaction half-life is greater than 30 sec, even for the fastest conditions considered here. Reference (*1*) provides details of the general-purpose interfacing capabilities of the PRTB system which allow synchronization of experimental and computer functions.

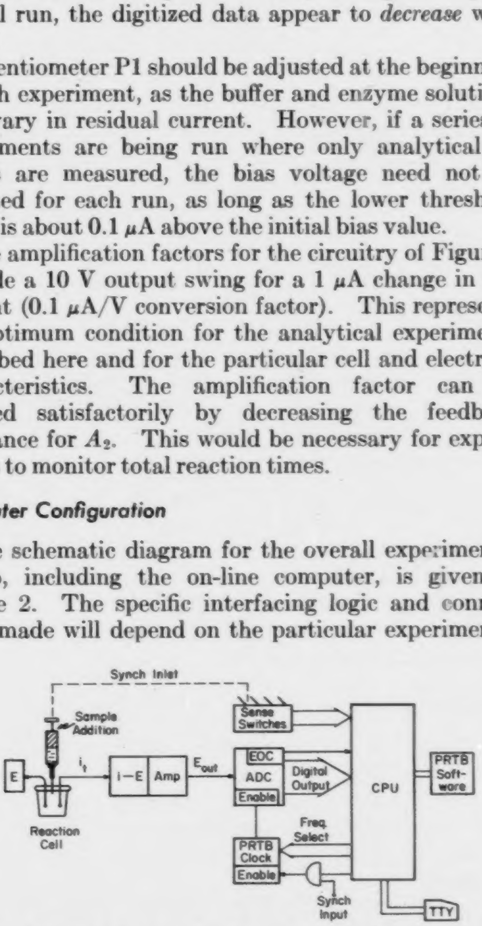


Figure 2. Schematic diagram of on-line reaction rate instrumentation.

Procedure

The detailed procedure presented here is for experiments where the relationship between G_0 and $1/\Delta t$ for a specified current interval, Δi, is determined and used for quantitative analysis. (The fundamental equations are provided in the Introduction.) Basically, the experiments involve establishing an interval, Δi, and programming the computer to monitor the output signal to determine Δt required to traverse the lower and upper signal thresholds bracketing Δi. Procedural details are as follows.

All reagents, standards, and samples are adjusted to the working temperature by immersion in a water bath. The sample compartment is rinsed with de-ionized water and a siphon is used to remove the water from the compartment. One milliliter of composite reagent is added by hypodermic syringe to the cell and the stirrer is turned on. The power supply to the monitoring circuit is then turned on, and after about 5 sec, the initial voltage will have stabilized. (NOTE—The electrodes should be polarized before use, or, alternatively, several trials can be performed before activating the computer. Otherwise, the initial voltage set by P1 in Figure 1 will not remain stable for the first few trials.) Adjust P1 to bring the voltage to about −9.5 V. Initiate the computer program. Add 1.00 ml of sample to the cell. Use a manual sense switch to indicate to the computer when it should begin to monitor the output signal. The computer might be programmed to count CLOCK pulses while the output signal traverses the region between a lower and upper threshold specified in the program (Δi). When the upper threshold is surpassed, the computer could print out the trial number, the time, and the inverse time and ask if another trial is to be attempted. Before executing another run, the user should turn off the circuit power supply, turn off the stirrer, and then drain and rinse out the reaction cell. [NOTE—The initial adjustment of P1 could be checked before re-running the experiment by programming the computer to sample and print the initial output bias voltage.]

A ten Hz clock rate gives sufficient accuracy when the averages of at least three trials for each standard or unknown are used. Different clock rates can be tried to see if any increase in accuracy and precision is obtained.

Since a 1.0°C temperature rise causes a 10% rise in the reaction rate [ref. (*6*)], the water bath temperature should be kept to ±0.1°C to ensure ±1% accuracy.

Reagents

All solutions in the experiment are prepared in de-ionized water and are stored in a cold room at 5°C.

Buffer-catalyst. The buffer-catalyst is prepared by dissolving 82 g of potassium dihydrogen phosphate, 42 g of potassium monohydrogen phosphate, and 13 g of ammonium molybdate ($(NH_4)_6Mo_7O_{24} \cdot 4H_2O$) in de-ionized water and diluting the solution to a volume of one liter. This solution is stable indefinitely at 5°C.

Potassium Iodide. Dissolve 83 g of potassium iodide in de-ionized water and dilute to 1 l. This solution is stable for several weeks when stored at 5°C.

Glucose Oxidase. Dissolve 0.06 g of glucose oxidase (Sigma Type II) in 50 ml of de-ionized water. This solution is stable for one to two weeks at 5°C.

Composite Reagent. Fifty milliliters of composite reagent are prepared by mixing 20 ml each of both the buffer-catalyst and potassium iodide solutions and 10 ml of the enzyme solution. This solution should be stored in a flask which is protected from the light.

Glucose Standards. Standard glucose solutions of 10, 25, 50, and 100 are prepared by the dilution of a 1000 ppm glucose solution prepared by dissolving 1.000 g of C.P. reagent sugar in de-ionized water and diluting to 1 l. These standard solutions are stored at 5°C.

Discussion

The kinetic analysis experiment described here has been run by students in Junior/Senior level analytical courses at Purdue. An approach will be described here for an experiment oriented towards quantitative analysis. Typically, programs have been written which provide the following functions: Data acquisition from kinetic runs on standard samples; least squares fitting of a linear calibration function; data acquisition from unknown samples; and print-out of analytical results. In addition, several options might be included in the on-line program—such as the possibility of averaging a series of runs before analytical computations are made, rejecting any given run, or determining a new calibration curve. The program could also allow for selection of the basic CLOCK frequency and Δi interval.

A flowchart of a program corresponding to the above description is provided in Figure 3a. Figure 3b shows the details of the data acquisition flowchart segment. Figure 4 gives a listing of the data acquisition subroutine written in the PRTB format (*1*). [The reader should refer to reference (*1*) for a detailed description of the PRTB CALL functions. However, to aid the reader in interpretation of this program segment, the CALL statements used are briefly defined here.] CALL (1,X,F) initializes the data acquisition hardware and software, specifies the data variable, *X*, and the CLOCK frequency, *F;* CALL (2) starts the CLOCK; CALL (3) takes in a digitized datum at time intervals determined by the CLOCK frequency; and CALL (10) turns off the CLOCK. The subroutine serves to monitor the output signal (*X*), to compare it to the lower and upper thresholds (*A* and *B*), and to measure the time interval to traverse *A* and *B* (T1/F). (Note that the directions of the inequality signs in statements 625 and 640 reflect the fact that the *digitized* values *decrease* during the experiment, as pointed out in the Experimental section. If the digital values were *increasing* with time, the inequality signs should be reversed.) Because the noise level of the output signal was <1%, there was no need to incorporate noise-rejection steps in the data acquisition program. If noise were a problem, an averaging algorithm could be used when monitoring the data.

Example computer/operator dialog and results print-out are shown in Figure 5. Typical analytical results over the range 10–100 ppm glucose are also given in Figure 5. A typical calibration plot for this system is given in reference (*8*). A distinct non-zero intercept is normally observed, fixing the lower limit of sensitivity to 10 ppm.

Suggested Experimental Assignments

Once the kinetic analysis cell, monitoring circuitry, and on-line computer system have been set up, a variety of experimental studies can be made. Moreover, because the computer reduces considerably the data handling and processing chores, the student has time to investigate several aspects of the experimental system. Listed here are several options suggested

(1) Quantitative analysis, as outlined above

```
600  LET T1=0
605  CALL (1,X,F)
607  PRINT
610  PRINT "READY TO BEGIN EXPERIMENT"
615  CALL (2)
620  CALL (3)
625  IF X>A THEN 620
630  LET T1=T1+1
635  CALL (3)
640  IF X>B THEN 630
645  LET T1=T1/F
650  CALL (10)
655  RETURN
```

Figure 4. Listing of data acquisition subroutine only.

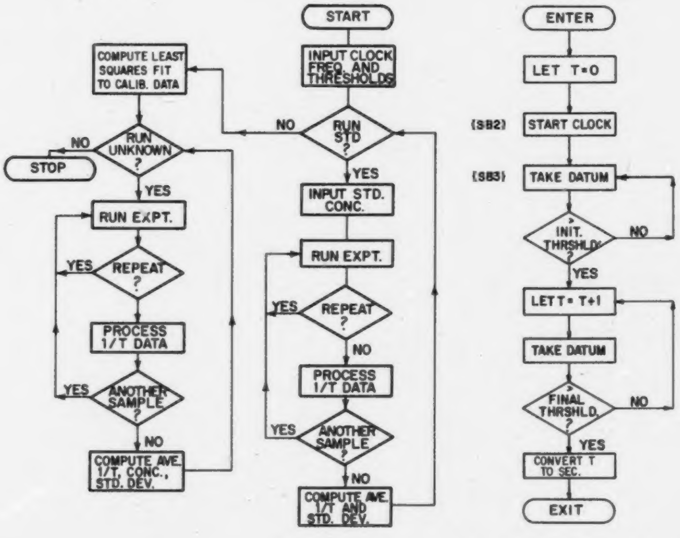


Figure 3. a, Overall flowchart for kinetic analysis experiment. b, Flowchart for data acquisition program segment.

```
RUN
THIS PROGRAM ANALYZES THE KINETICS OF THE OXIDATION OF
     GLUCOSE BY THE ENZYME GLUCOSE OXIDASE.

FOR ALL YES/NO QUESTIONS WHICH THE USER OF THIS PROGRAM
     IS ASKED, TYPE 1 FOR YES AND 0 FOR NO.  FOR ALL
     OTHER QUESTIONS, TYPE IN THE NUMERICAL VALUE OF
     THE QUANTITY BEING SOUGHT.

WHAT ARE THE INITIAL AND FINAL THRESHOLDS FOR ALL RUNS?700,200

DO YOU WANT TO RUN A STANDARD?1
WHAT IS THIS STANDARD CONCENTRATION (IN PPM)?100

READY TO BEGIN EXPERIMENT
CONC,= 100     TRIAL 1       T= 6.7              1/T= .149254
DO YOU WANT TO REPEAT THIS TRIAL?0
ANOTHER TRIAL?1


READY TO BEGIN EXPERIMENT
CONC,= 50      TRIAL 3       T= 14.4             1/T= 6.94444E-02
DO YOU WANT TO REPEAT THIS TRIAL?0
ANOTHER TRIAL?0
CONC.= 50      AVERAGE 1/T FOR 3  TRIALS = 6.88124E-02
     STD. DEVIATION= 7.20887E-04


DO YOU WANT TO RUN A STANDARD?0
LEAST SQUARES FITTED LINE IS --
     1/T= 1.44942E-03   C +-4.38669E-03
     1/T IN SEC AND C IN PPM

CONC.           AVE. 1/T       DEVIATION
 100             .140376        1.78814E-04   ( .13%)
 50             6.88124E-02   -7.28279E-04   (1.1%)
 25             3.09792E-02    8.69516E-04   (2.8%)
 10             1.04274E-02   -3.19956E-04   (3.1%)

DO YOU WANT TO RUN AN UNKNOWN?0
RECORD VALUES IN  NOTEBOOK.  GOODBYE.
```

Figure 5. Samples of computer/operator dialog. (Note: Segments are sampled from a much longer total dialog. Also % deviations added later.)

(2) Kinetic studies to determine reaction orders and rate constants

(3) Investigation of effects of experimental parameters on accuracy and precision of quantitative data. (e.g., Δi and CLOCK rate can be varied; temperature, *p*H and solvent effects can be studied.) For a description of the various features of this system open to fundamental study, refer to the original literature (*6–8*).

Literature Cited

(1) Perone, S. P., and Eagleston, J. F., J. Chem. Educ., **48,** 317 (1971).
(2) Perone, S. P., and Eagleston, J. F., J. Chem. Educ., **48,** 438 (1971).
(3) Toren, E. C., J. Chem. Educ., **44,** 172 (1967).
(4) Pardue, H. L., *Rec. Chem. Progr.*, **27,** 151 (1966).
(5) Guilbault, G. G., *Anal. Chem.*, **38,** 527R (1966).
(6) Malmstadt, H. V., and Pardue, H. L., *Anal. Chem.*, **33,** 1040 (1961).
(7) Pardue, H L., *Anal. Chem.*, **35,** 1241 (1963).
(8) Pardue, H. L., Burke, M. F., Jones, D. O., J. Chem. Educ., **44,** 684 (1967).

F. G. Pater
and S. P. Perone
Purdue University
Lafayette, Indiana 47907

Computer-Controlled Colorimetry

An undergraduate experiment

In an effort to provide undergraduates with experience in on-line experimentation in the chemistry laboratory, several laboratory exercises have been developed at Purdue. These include data acquisition from synthetic experimental signals *(1, 2)*, a gas chromatographic experiment *(3)*, and a kinetic enzymatic analysis experiment *(4)*. The programming language utilized is a modified BASIC (referred to as Purdue Real-Time BASIC, PRTB) which allows data acquisition and control commands *(2)*. In these experiments, the computer has been a very powerful and convenient data collector and processor. However, the use of the computer not only for data acquisition but also for experiment control is feasible and desirable. In order to illustrate more fully the capabilities of the computer for experimental control, a computer-controlled colorimeter was conceived, where the computer could select the wavelength of light passing through the sample as well as execute data acquisition and data processing functions.

Approach and Design

The choice of a Bausch and Lomb Spectronic 20 as the instrument to automate was made on the basis of its availability, low cost, and simple, rugged design. Since only a moderate scan speed was desired and high resolution was not necessary, a synchronous motor was employed to control the dispersive element in the colorimeter. To achieve a more drift-free signal from the phototube, it was decided to bypass completely the standard amplifiers in the instrument and substitute a solid state operational amplifier to provide the data signal to the computer.

The computer instrumentation consists of the General Purpose Interface (GPI) hardware described elsewhere *(2)* and a Hewlett-Packard 2116A computer with 8K of core memory, a high-speed paper tape reader and punch and a Tektronix Model 601 storage oscilloscope. An interface to the GPI was constructed to translate computer outputs to mechanical actions controlling the monochromator. This interface to the GPI is shown in Figure 1.

In the control of any motor, two signals are inherent: a direction of rotation signal, and a magnitude of rotation signal. Since a synchronous motor was used, the amount of rotation is determined by the time the motor is allowed to rotate. Starting and stopping the motor are accomplished through activation of a clutch on the armature. Two control signals are required and are used to actuate transistor relay drivers. These relays in turn start and stop the motor (engage or disengage the clutch) and determine the direction of rotation.

Because the computer is controlling the rotation of the grating in the Spectronic 20, it must also be able to sense the position of the grating. Thus, a potentiometer has been connected to the shaft of the monochromator through gears mounted on the monochromator shaft and on the potentiometer wiper. As the monochromator shaft is rotated, the potentiometer changes its resistance and acts as a voltage divider. Because the angle of rotation of the grating is linear with wavelength, the fraction of the applied voltage appearing at the armature of the potentiometer is related to the wavelength of light passing through the cell. This analog signal must be digitized through the analog-to-digital converter (ADC). Because phototube data must also be acquired through the same ADC, these two signals (wavelength sense and phototube output) must be multiplexed. In order to control the multiplexer, two more digital commands must be available to select the appropriate multiplexer channel. Thus, for control of the experiment, a total of four digital signals or bits are required: two for motor control and two for data acquisition control.

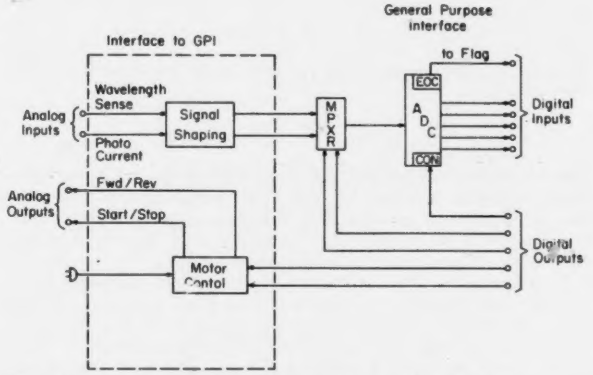


Figure 1. Block diagram of the interface between the Spectronic 20 and the computer.

Experimental

Hardware

Figure 1 shows a block diagram of the interface. The block called MOTOR CONTROL consists of the two relay drivers and relays shown in Figure 2. The motor which they control is a 4 rpm synchronous motor (Model AR-DA, Hurst Co., Princeton, Ind.). The multiplexer and ADC are available on the GPI and have been described elsewhere *(2)*.

The signal shaping part of the interface is shown also in Figure 2. The signals to the multiplexer are the two data signals discussed previously. The analog input from the potentiometer is a varying voltage and is followed by an operational amplifier (Fairchild 741) (A2) used in a voltage-follower mode. The other signal, the photocurrent, is taken directly from the phototube. Another operational amplifier (Fairchild 741) (A1) is used to convert the photocurrent to a voltage capable of being converted by the ADC. The capacitor C_1, in Figure 2, is used to limit the bandpass of this amplifier. The resistor network consisting of resistors R_2, R_3, R_4 is used to compensate for any dark current from the phototube. R_1, the feedback resistor, was chosen so that a photocurrent of approximately 150 nanoamps will produce a -10 V output from the current-to-voltage converter. (The ADC input range is ± 10V.)

Operational Features

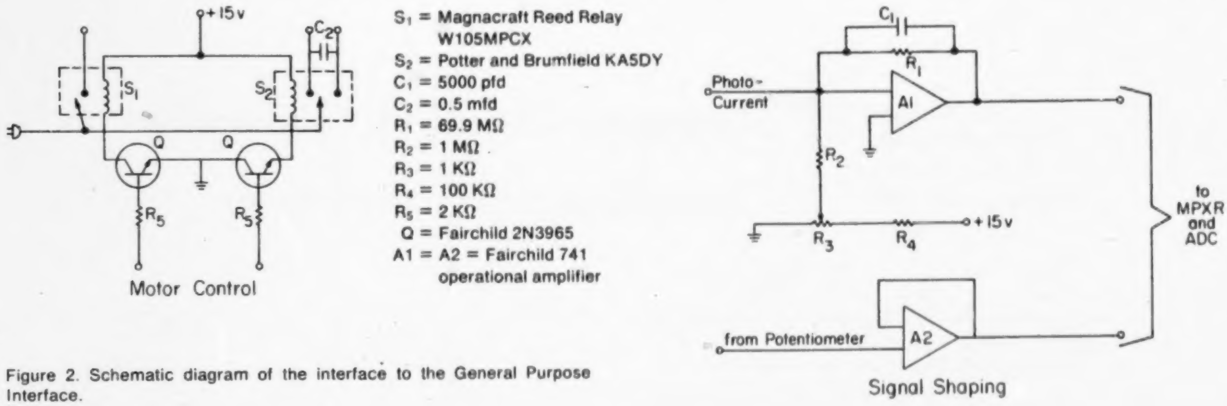
The software required to operate this modified Spectronic 20 will need to supervise data acquisition as well as control the wavelength. For data acquisition, the computer must decide which datum (wavelength or transmittance) to take at any given moment in real-time, select the proper multiplexer channel, and acquire the datum.

If the datum is from the phototube, then no real-time calcula-

---

This work supported by a grant from the National Science Foundation Office of Computing Activities, Grant No. GJ-428.

[Reprinted from Journal of Chemical Education, Vol. 50, Page 428, June, 1973.]
Copyright, 1973, by Division of Chemical Education, American Chemical Society, and reprinted by permission of the copyright owner


+15v
C2
S1
S2
Q
Q
R5
R5
Motor Control
C1
R1
Photo-Current
A1
R2
R3
R4
+15v
to MPXR and ADC
from Potentiometer
A2
Signal Shaping

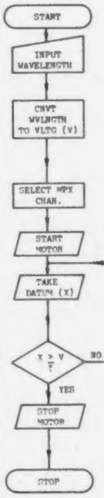
S1 = Magnacraft Reed Relay W105MPCX
S2 = Potter and Brumfield KA5DY
C1 = 5000 pfd
C2 = 0.5 mfd
R1 = 69.9 MΩ
R2 = 1 MΩ
R3 = 1 KΩ
R4 = 100 KΩ
R5 = 2 KΩ
Q = Fairchild 2N3965
A1 = A2 = Fairchild 741 operational amplifier

Figure 2. Schematic diagram of the interface to the General Purpose Interface.

tions on it are required although some may be done. For example, if one wished to subtract out the blank values in real-time, this may be done as long as the data acquisition rate is slow enough. Data from the phototube need to be corrected for blank values due to the non-linear response of the phototube.

If the datum is from the monochromator, the computer must decide in real-time if the proper wavelength has been reached and output the proper control bits to allow the motor to continue or to stop it. In this case, the data rate must be slow enough to allow for whatever real-time processing is necessary to determine the position of the grating. Since the motor is relatively slow (4 rpm), the entire spectrum (300 nm) is scanned in approximately 6 sec. If a resolution of one part in 300 is required, then the ADC must sample at least 300 data in those 6 sec or 50 points/sec. With these specifications in mind, a flow chart (Fig. 3) can be written for a program to set the monochromator to any wavelength desired. Note that some real-time processing is necessary. At a 50 Hz data rate, this processing can take no longer than 20 millisec which is time for the execution of perhaps two or three BASIC statements.

In order to demonstrate the usefulness of this system and to illustrate the principles involved in the software necessary to operate the Spectronic 20 under computer control, a program which allows this instrument to be operated as a pseudo-double beam recording spectrophotometer can be written in PRTB. A flowchart for such a program is given in Figure 4. The program corresponding to this flowchart is available from the authors upon request. The instrument becomes a pseudo-double beam recording spectrometer in this manner: the blank solution is scanned, and data are taken from the phototube at some fixed rate. The digitized values are saved in an array B(I). Next, the sample is placed in the colorimeter and the spectrum scanned at the same data rate as the blank. These data are saved as another array X(I). After data acquisition is complete, the data are converted to % transmittance and can be listed or plotted on the Teletype or the storage oscilloscope.


START
INPUT WAVELENGTH
CNVT WVLNGTH TO VLTG (V)
SELECT MPX CHAN.
START MOTOR
TAKE DATUM (X)
X ≥ V ?
NO
YES
STOP MOTOR
STOP

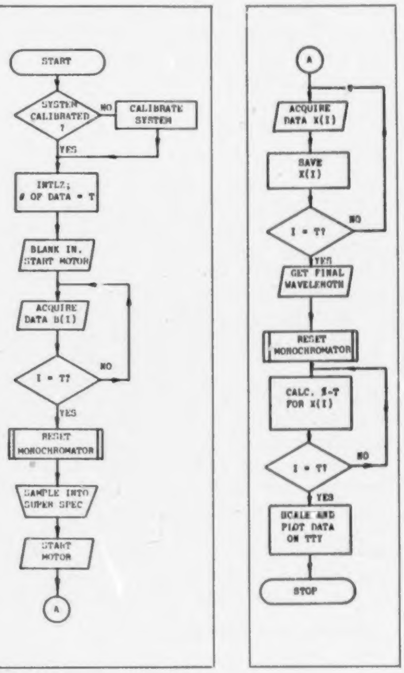

START
SYSTEM CALIBRATED ?
NO
CALIBRATE SYSTEM
YES
INTLZ; # OF DATA = T
BLANK IN. START MOTOR
ACQUIRE DATA B(I)
I = T?
NO
YES
RESET MONOCHROMATOR
SAMPLE INTO SUPER SPEC
START MOTOR
A
A
ACQUIRE DATA X(I)
SAVE X(I)
I = T?
NO
YES
GET FINAL WAVELENGTH
RESET MONOCHROMATOR
CALC. %-T FOR X(I)
I = T?
NO
YES
SCALE AND PLOT DATA ON TTY
STOP

Figure 3. (left) Flow chart for a program to set the monochromator to any wavelength requested.

Figure 4. (right) Flow chart for a program to operate the Spectronic 20 as a scanning spectrophotometer.

Discussion

With this system, a number of experiments in the undergraduate chemistry laboratory can be envisioned. (1) Through computer control a simple colorimeter can be used as a pseudo-double-beam spectrophotometer, as above. (2) Since data acquisition rates can be quite high if no real-time processing is involved (up to 20 KHz), time-dependent analysis of absorption peaks can be done for kinetics studies. (3) On a slightly slower time scale, time-dependent absorbance studies at multiple wavelengths can be done utilizing computer control of the monochromator. (4) Multiple data bracketing peak regions can be acquired, thus insuring that analytical measurements are made at peak maxima. (5) For very weakly absorbing systems, signal averaging can be performed.

One experiment developed here involves the use of Job's Method *(5, 6)* (the method of continuous variations) to elucidate the composition of some complexes formed between Ni(II) and ethylenediamine. The computer-controlled colorimeter is used to minimize the number of manual adjustments usually required, relieve the tedium of the experiment, and provide more effective data collection and handling. In addition, the computer can be programmed to process and display the data. (A detailed write-up providing the background and directions for this experiment is available from the authors on request.) An important observation here is that the addition of an experiment involving direct computer control exposes students to the important principles of automation and optimization of instrumental measurements.

Literature Cited

(1) Perone, S. P., J. CHEM. EDUC., **47**, 105 (1970).
(2) Perone, S. P., and Eagleston, J. F., J. CHEM. EDUC., **48**, 317 (1971).
(3) Perone, S. P., and Eagleston, J. F., J. CHEM. EDUC., **48**, 438 (1971).
(4) Jones, D. O., Scamuffa, M. D., Portnoff, L. S., and Perone, S. P., J. CHEM. EDUC., **49**, 717 (1972).
(5) Job, P., *Ann. Chim.* [10]**9**, 113 (1928).
(6) Vosburgh, W. C., and Cooper, G. R., *J. Amer. Chem. Soc.*, **63**, 437 (1941).

FUNCTION GENERATOR #1
(SINE-EXPONENTIAL GENERATOR)

The sine and exponential wave generator was developed for use within several of the Chemistry computer experimental courses. The sine wave generator is a modified amplitude controled Wien bridge oscillator with a buffered output, while the exponential generator consists of two FET (field effect transistor) switches which alternately charge and discharge a timing capacitor.

Two trim pot adjustments are available for the sine wave generator. R_B (50K) is adjusted for oscillation and a general range of amplitude while R_A (5K) is used to trim the amplitude to the desired magnitude. The two adjustments interact and R_B should be adjusted first.

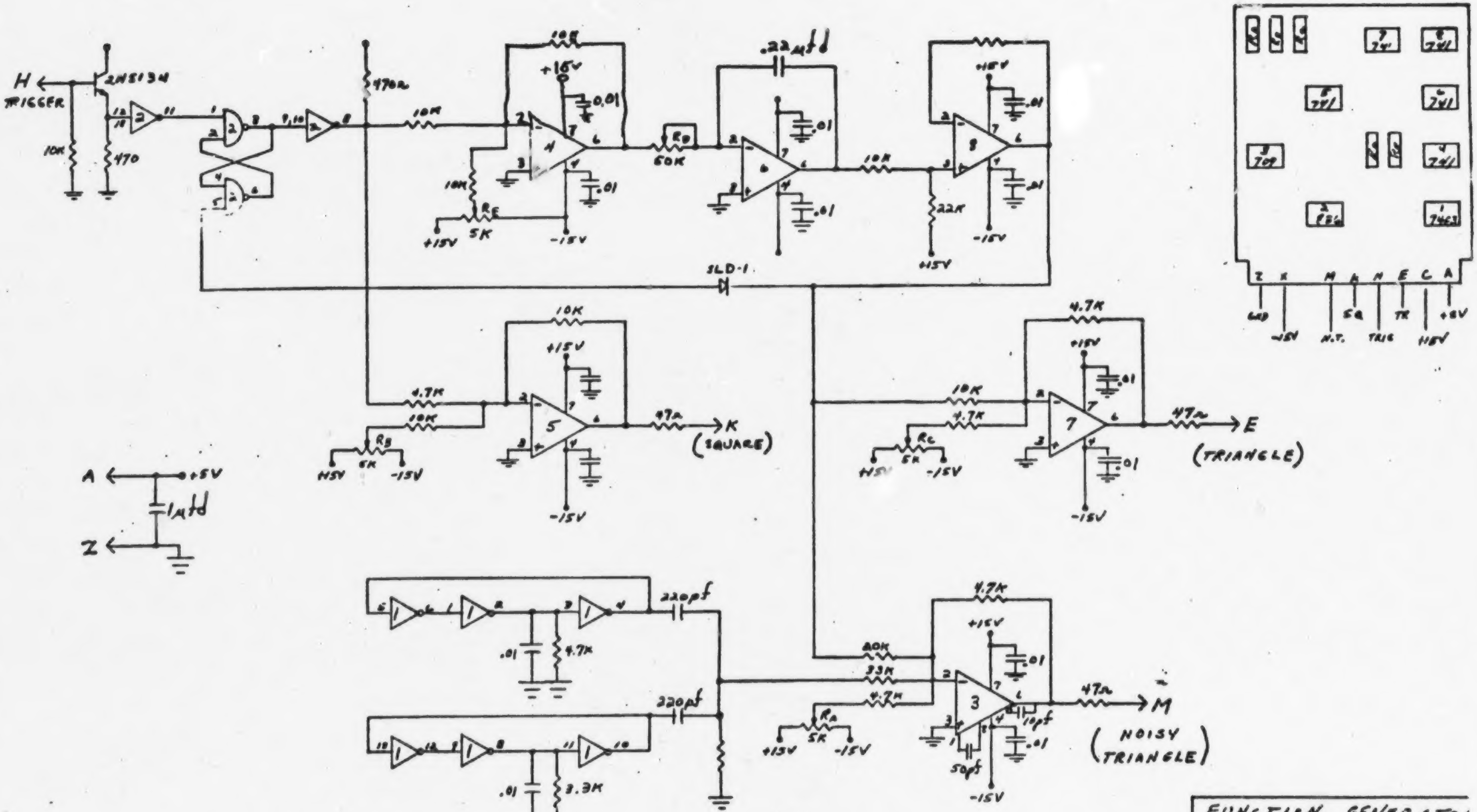
Capacitors C_1 and C_2 are chosen to approximate a .1 second time constant with R_D the output resistor and the series resistance of the FET switch 2. The charging time constant determined by C_1, C_2, R_C and the series resistance of FET switch 1 is less providing for a quicker reset condition.

With the input gate signal low switch 1 is on permitting C_1, C_2 to charge through R_C and the R_{ON} of the switch to the potential level established by the zener diode. On the alternate half cycle when the input gate signal is high switch 2 is on and now C_1 and C_2 discharge through R_D and the R_{ON} of the switch. The gate signal duration should be at least .4 seconds for output and .3 seconds for reset.

The output signal is approximately 8 volts in magnitude and total decay time of between .3 and .4 seconds as indicated below.

Pin connections for this card are listed below and shown on drawing 114-01 along with a complete schematic.

pin	function
1	+5 Volts
3	+15 Volts
5	Exponential output
7	Gate for exponential
18	Sine wave output
20	-15 Volts
22	Ground


H
TRIGGER
2N5134
10K
470
470Ω
10K
10K
+15V
0.01
RE
5K
+15V
-15V
.01
RD
60K
.22μfd
.01
10K
22K
+15V
-15V
.01
SLD-1
10K
+15V
4.7K
10K
RB
5K
+15V
-15V
47Ω
K
(SQUARE)
4.7K
10K
4.7K
RC
5K
+15V
-15V
.01
47Ω
E
(TRIANGLE)
A
+5V
1μfd
Z
220pf
220pf
.01
4.7K
.01
3.3K
4.7K
80K
33K
4.7K
RA
5K
+15V
-15V
.01
10pf
50pf
47Ω
M
(NOISY TRIANGLE)
741
709
8E6
74C3
Z X M L N E C A
GND
-15V
N.T.
SQ
TRIG
TR
+15V
+5V

FUNCTION GENERATO

SQUARE, TRIANGLE, NOISY T

DRAWN BY ECS — DATE 5/5/72

APPR ECS — DATE

114-0.

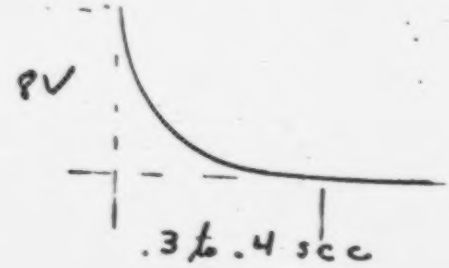
FUNCTION GENERATOR #2
(SQUARE, TRIANGLE, NOISE TRIANGLE)

This function generator card was implemented for use in several of the Chemistry computer experimental courses. The function generator is a copy of the generators developed earlier by D. O. Jones and described in the article "Synthetic Signal Sources for Computer Experiments in the Undergraduate Chemistry Laboratory". Several additions have been included providing for quick adjustment.

Drawing number 114-02 shows the generator in its present form. The units are constructed on standard general purpose "Vero P. C." board and are wired with the connections as shown below.

pin number	description
A	+5 Volts
C	+15 Volts
E	TRIANGLE
H	TRIGGER
K	SQUARE
M	NOISY TRIANGLE
X	-15 VOLTS
Z	GROUND

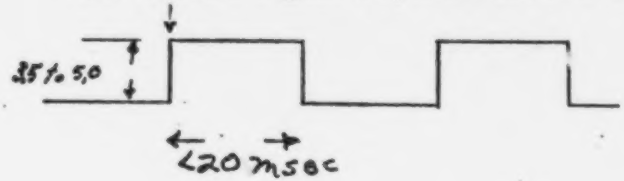
Five adjustment pots exist on the present cards and these pots are labeled R_A, R_B, R_C, R_D and R_E on the drawing. Proper adjustments were made at the time of construction and if any further changes are necessary refer to the listing below.

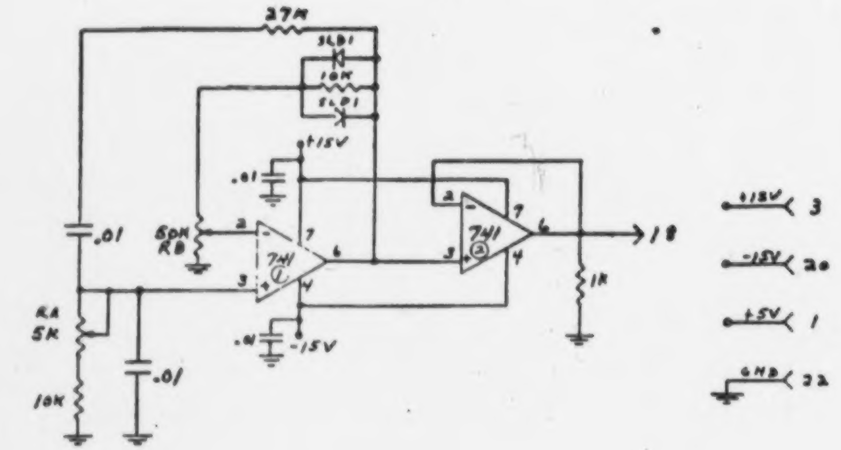


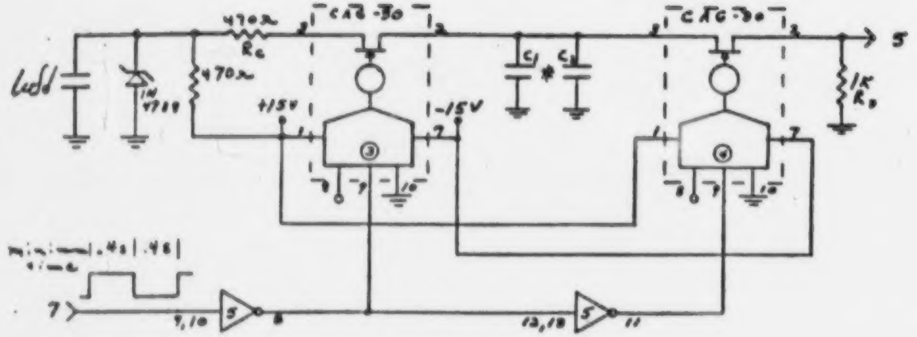
R_A	OFFSET FOR NOISY TRIANGLE
R_B	OFFSET FOR SQUARE WAVE
R_C	OFFSET FOR TRIANGLE
R_D	INTEGRATION TIME CONSTANT
R_E	TRIANGLE RISE

R_D and R_E are adjusted together to produce a symmetrical triangle.

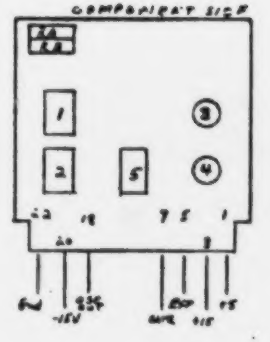
The TRIGGER signal requirements are shown below.




27M
3LD1
10K
3LD1
+15V
.01
50K
RB
741
①
741
②
18
1K
RA
5K
.01
.01
.1
-15V
10K
+15V 3
-15V 20
+5V 1
GND 22


470Ω
Rc
CAG-30
CAG-30
5
C1
C2
*
1K
Ro
470Ω
1N4729
+15V
-15V
③
④
minimal .4s .4s
time
7
9,10
5
8
12,13
5
11

*SELECT C1, C2 for
TOTAL EXPONENTIAL .3 SECONDS


COMPONENT SIDE
1
2
5
③
④
22
18
20
7
5
3
1
Gnd
-15V
-15V
OUT
GATE
+15
+5

IC 5 SN7400

SINE - EXPONENTIAL GENERATOR

DRAWN BY
DATE 4/27/72
APPR
DATE
114 -01

LOGIC EXPERIMENT I

INTRODUCTION TO DIGITAL LOGIC

Purpose.

This experiment is designed as an introduction to digital logic and to the El Instruments Elite 3 logic lab. It introduces the usage of integrated circuit logic devices and their application in solving various logic and gating problems and in constructing a number of common flip-flop and counting circuits.

Equipment.

1. Elite 3 Circuit Design Test System
2. Various I.C. devices:
    - 4 SN7400 Quad 2-input positive NAND gates
    - 1 SN7402 Quad 2-input positive NOR gate
    - 1 SN7404 Hex inverter
    - 2 SN7410 Triple 3-input NAND gates
    - 1 SN7420 Dual 4-input NAND gate
    - 2 SN7473 Dual J-K Master-Slave flip-flops
    - 1 SN7401 Quad 2-input NAND gate (with open collector output)
    - 3 1.5 KΩ resistors

Revised, April, 1973

The Logic Laboratory.

The logic lab is a device that provides a convenient means of constructing and testing complex digital or analog circuitry. It can be employed either as an educational aid or as a research tool, permitting the evaluation of complex circuitry before it is actually built into a system.

The Elite 3 makes use of a positive five volt logic system in which five volts is defined as logical "1" or "true" and 0 volts is considered a logical "0" or "false". Accordingly, a +5 volt source is provided by several terminals running along the two red lines at the top of the instrument, and ground (or common) is found at the bottom. A number of toggle switches are provided as convenient sources for logical "1" and logical "0" signals; and, in addition, there are four push-button switches which serve the function of manually operated, low frequency pulse generators. The remainder of the logic lab consists of a series of indicator lights and the breadboard on which the circuits are physically assembled.

In order to familiarize yourself with the instrument, be sure to verify the logic level required to light the indicator lamps, the manner in which the panel switches function, and the physical and electrical layout of the breadboarding. (*e.g.*, note that the terminals set in the Teflon blocks are arranged so that those in each horizontal row of a strip are connected together.)

Next hook up each of the I.C. devices provided for the experiment according to the data sheets and verify that they are functioning properly. Where necessary, use panel switches to vary input levels and panel lamps to monitor output logic levels.

CAUTION: DO NOT TURN ON THE POWER SWITCH OR SUPPLY POWER TO THE I.C. UNTIL YOU ARE CERTAIN OF CORRECT WIRING. ALSO, CONNECTION OF THE OUTPUTS OF TWO TTL DEVICES CAN DESTROY BOTH.

Make sure that you have checked the following:

(a) What level lights a lamp?

(b) Which panel terminals are connected in parallel?

(c) How do the panel switches function?

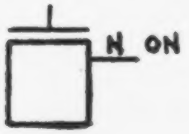
(d) How can power and ground be distributed?

(e) Verify the fact that open inputs on I.C.'s are logical 1's.

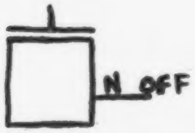
(f) Use the I.C. data sheets to guide hook-up of each device.

(g) Connect the pulse generator "High" output to the Logic Lab; connect the pulse train output to the C input of a JK Flip-Flop and monitor the Q output as you vary the pulse frequency on the low range.

NOTE: In this and later experiments, the four push-button switches will be referred to as pulsers and will be represented diagramatically as follows:



For a pulser that is normally in the "1" state. When depressed, a logical "0" is output.

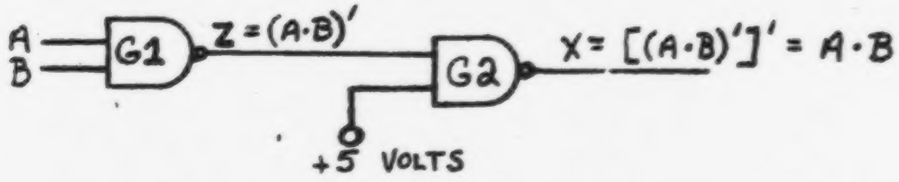


For a pulser that is normally in the "0" state. When depressed, a logical "1" is output.

OPTIONAL - Logic and Gating Problems.

The set of I.C. devices that you have been given consists primarily of 2-, 3-, and 4-input NAND gates. But from these alone can be generated most simple logic functions. A 3-input NAND gate can, for example, be used to simulate a 2-input gate. Using truth tables and/or Boolean expressions, decide how you would handle the unused input and confirm your answer by comparing the modified 3-input gate with the 2-input gate. Similarly, design a circuit that uses a 2 (or more) input gate to invert a logical signal and again confirm your design experimentally. Remember that open circuit inputs to TTL gates represent logical "1" states.

These latter experiments were included to demonstrate the characteristics of these gates. These characteristics must be kept in mind when using several gates to implement a more complex Boolean expression. For example, suppose you wanted to use 2-input NAND gates to implement the AND function $X = A \cdot B$. You note that a single NAND will generate the function $Z = (A \cdot B)'$; and, therefore, that the desired function X can be generated by inverting Z, ($X = Z'$ or $X = [(A \cdot B)']' = A \cdot B$). This function is implemented as follows:



Gate G1 performs the NAND function, and G2 changes the sense of or inverts the output from G1.

As an exercise, implement a 3-input NOR function using only NAND gates. Note that the expression to be implemented is:

$$Z = \overline{A+B+C}$$

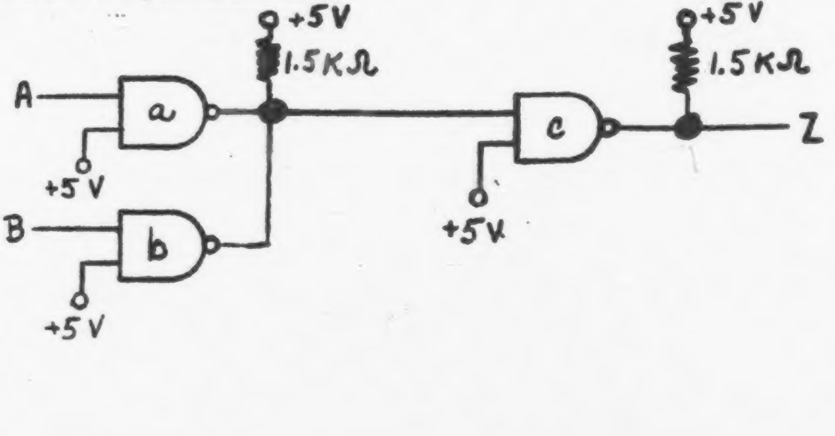
which can be complemented by means of De Morgan's Theorem to yield:

$$Z = \overline{A} \cdot \overline{B} \cdot \overline{C}$$

Wire up this function and confirm that it performs the required NOR by constructing a truth table.

In an analogous manner, implement the expression Z = A·B+C·D using NAND gates. Compare your results with the truth table for the expression.

In addition to the usual SN7400 NAND gate that you have been using up to this point, there is another type of NAND gate, the SN7401 NAND gate with an open collector output. The primary difference between the two is that, while two normal TTL gates cannot be safely used with their outputs directly connected, two open collector type I.C.'s can. As a result, open collector gates allow you to construct several functions which would be either impossible or inconvenient to wire with the normal TTL gate. One such circuit is pictured below:

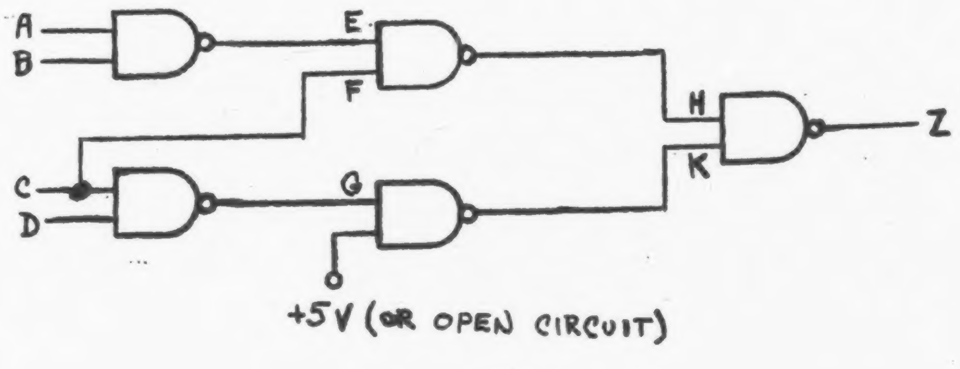


The truth table for this circuit is:

A	B	Z
0	0	0
1	0	1
0	1	1
1	1	1

The resulting logical function can be seen to be a simple OR function. Since it is constructed by wiring the outputs of two gates together, it is commonly called a wired-OR function. Construct this circuit where gates a and b and c are taken from an SN7401 I.C. Observe whether the behavior is as predicted above.

As you have by now discovered, the use of Boolean Algebra and truth tables is intimately related to the use and analysis of logic circuits. In fact, Boolean expressions, truth tables, and logic circuits are just different ways of saying the same thing and are completely interconvertible. Accordingly, the following circuit can be represented by a Boolean expression. Write the corresponding expression and, if you wish, test your analysis experimentally.



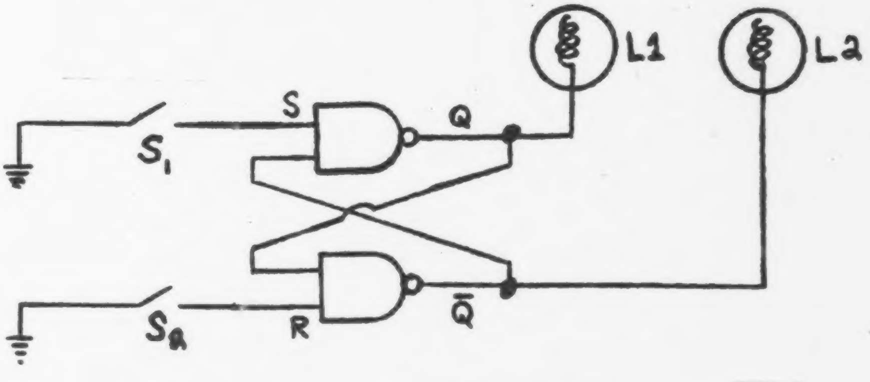
Finally, write the Boolean expression and design the logic for the following truth table:

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	0

Incidentally, the logic function represented by the truth table above is quite common and is termed the XOR (Exclusive OR) function.

REQUIRED - Flip-Flops.

The simplest type of flip-flop is the Reset-Set (RS) flip-flop which can be constructed from a pair of 2-input NAND gates as shown below:



RS Flip-Flop

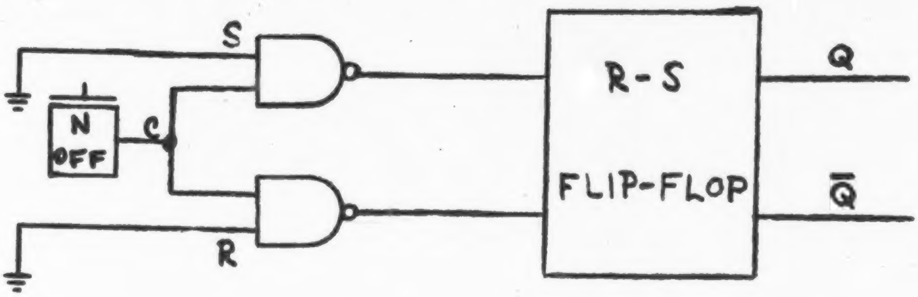
The inputs are labeled S for Set and R for Reset. It should be noted that the normal state of this flip-flop has both inputs at logical "1" (+5V or open circuit for positive logic in these experiments). The outputs are labeled Q and $\overline{Q}$. Wire the circuit and note the states of L1 and L2. One light should be on, and the other should be off. Using the truth table for a NAND gate, predict what, if anything, will happen if the switch corresponding to the indicator which is on (S1 for L1 and S2 for L2) is closed momentarily (a logical "0" condition). Try it. If your interpretation is correct, continue; if not, correct your interpretation. Now predict what will happen if the switch corresponding to the indicator which is off is closed momentarily. Try it.

The key feature of the RS flip-flop is that it has two stable states and that, once it is forced into one of these states, it will remain in that state until it is forced out of it. In other words, a single signal of short duration applied to the appropriate input will change the flip-flop to a known steady state.

Noting the above, is it necessary to have an indicator light on each of the flip-flop outputs? Also, can you predict the final state of the flip-flop if logical "0" is applied to and removed from both the R and S inputs simultaneously? The answer is No. Explain this observation by setting up a truth table for the R-S flip-flop.

There are a number of apparent disadvantages and limitations inherent in the design of the R-S flip-flop which impair its flexibility and utility. However, without fundamentally altering its design, a number of improvements can be obtained easily.

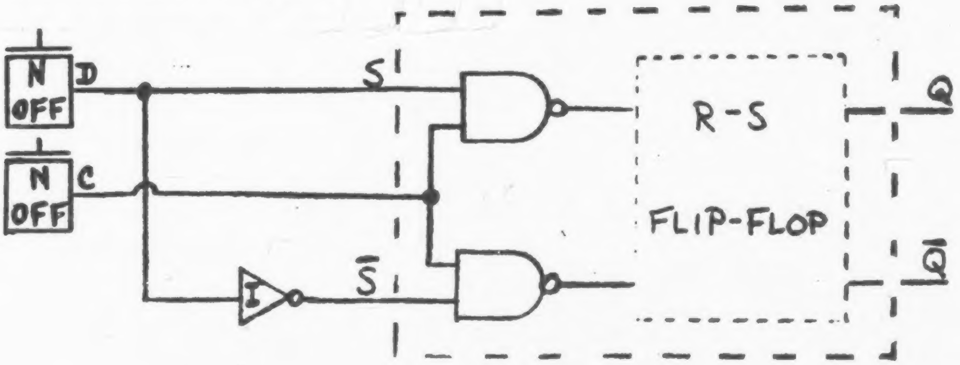
One simple improvement involves the use of gating at the input of the flip-flop. It is often desirable to synchronize the output of a control signal with the occurrence of one or more events. It may be desirable, for example, that the flip-flop be set only when a number of events (switch closures to ground, level changes, clock pulses, etc.) occur simultaneously. This can be accomplished by inserting NAND gates at the inputs as shown below:



Clocked RS FF

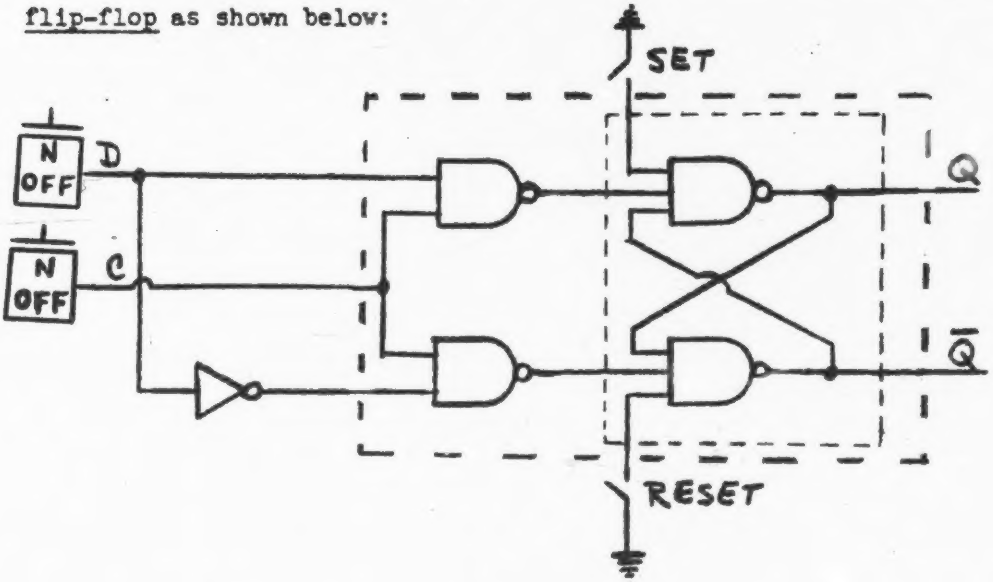
where S and R are the usual input signals and C represents the input from a switch, clock, or other element. Construct such a "gated" or "clocked" RS flip-flop in which C represents a switch normally closed to ground. Attempt to change the state of the flip-flop with the switch closed (logical "0"). Any success? Now try it with the switch open. It should be clear that if the switch were replaced by a clock, then information could be transferred to the flip-flop only when the clock pulse is true.

The clocked flip-flop just described required two inputs--one at S and one at R, to allow it to be set and reset. In many situations, it is desirable to have the flip-flop take on the state of a single data line when strobed by an enable pulse at C (from a clocked control circuit). A simple modification permits this:



Data Latch

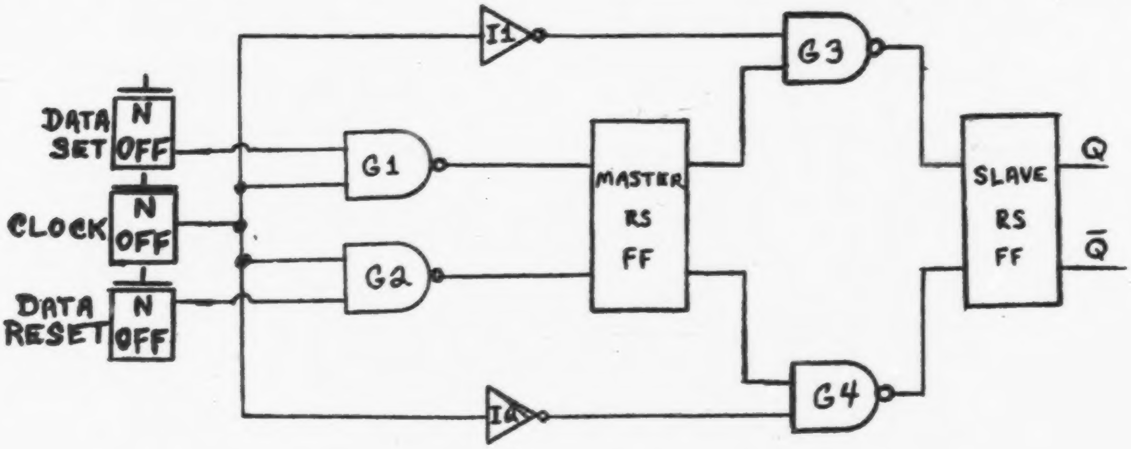
In many instances, it is also desirable to have direct Set and Reset (or Clear) inputs for the flip-flop. Yet, in its present state, there is no means of changing the state of the flip-flop except by entering information through the D input. This problem is resolved in the advanced data flip-flop as shown below:



Advanced Data FF

Wire this circuit up and test the reset. You should determine whether the status of the clock input has any effect on the use of the direct set and reset inputs. Also, if either the direct set or reset inputs is held at logical "0", can new data be strobed into the flip-flop to change its state? Explain this. Finally, it should be noted that, while the clock input is true, any change in the data input will be reflected immediately in a change in the output. However, the status of the data input at the instant of a "1" to "0" clock transition is preserved at the output as long as the clock stays false.

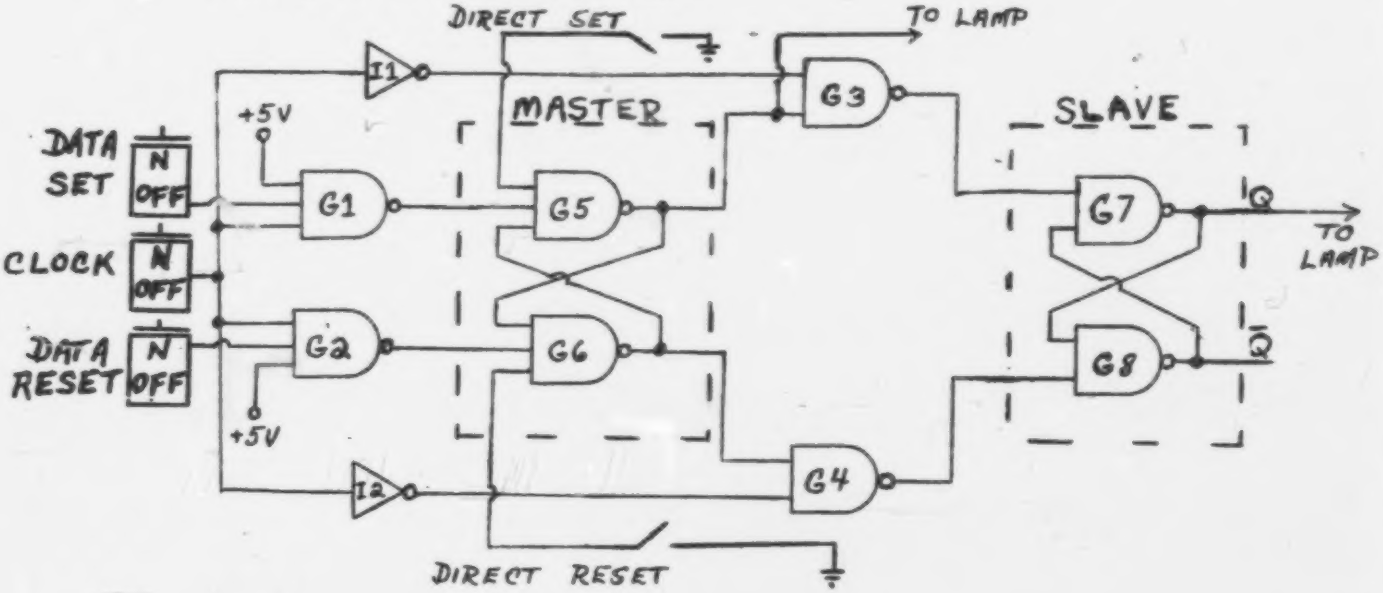
One of the most important applications of the flip-flop is in the construction of counters and registers. In many situations, it is desirable that a flip-flop accept and hold information at some time t while simultaneously maintaining information pertinent to time t-1 at its output. In other words, it is accepting current data from one device while it is transferring previously held data to another device. In most cases, a single clock pulse (level change) will trigger the flip-flop to accept data at the same instant it is triggering the next device to accept data from the flip-flop. In practice, this is accomplished by constructing the total flip-flop from two clocked RS flip-flops as represented below:



Master-Slave FF

The operation of this device is as follows: If either (not both) data input is true, then a positive level change (0 to 1) at the clock input will gate this information to the master flip-flop through G1 or G2 and set or reset this flip-flop depending upon whether the set or reset data line is true. The clock input is inverted by I1 and I2 to disable gates G3 and G4 so that the information at the output of the master flip-flop is blocked from the slave flip-flop as long as the clock input is true. However, when the clock input goes false, the inverters enable gates G3 and G4 to permit the output data from the master flip-flop to be entered into the slave flip-flop. In summary, then, information is entered into the master flip-flop on a positive-going level change at the clock input; and this information is transferred to the slave flip-flop on the negative-going level change at the clock input.

Construct the circuit shown below:


DIRECT SET
TO LAMP
I1
G3
MASTER
SLAVE
+5V
DATA SET
N
OFF
G1
G5
G7
Q
TO LAMP
CLOCK
N
OFF
DATA RESET
N
OFF
G2
G6
G8
Q̄
+5V
I2
G4
DIRECT RESET

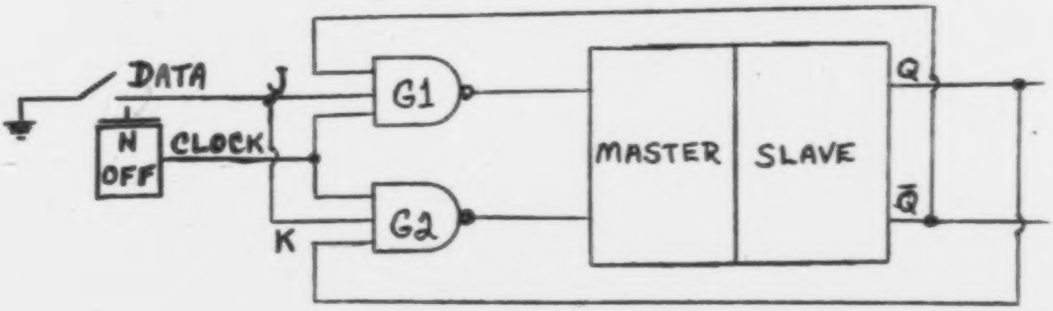
With the clock input at logical "0", confirm that the direct set and reset switches can be used to set and reset the flip-flop. Note also that with the clock input at this level, the slave flip-flop follows the state of the master flip-flop. Next enable the clock input (set to logical "1") and attempt to change the state of the flip-flop using the direct set and reset switches. Note that with the clock input enabled, the slave flip-flop cannot follow the master. Set the master flip-flop so that the master and slave are in opposite states (one indicator on, the other off); then disable the clock and note what happens to the slave flip-flop relative to the master.

Now clear the master flip-flop using the direct reset switch. Enable the data set input and observe the master and slave outputs. Any change? Leaving the data set input enabled, enable the clock input and observe the master-slave outputs. Any change? Disable the clock input and again observe the outputs. Any change? Enable and disable the clock input again. Any change?

At this point, you should have a reasonably good understanding of the operational features of the master-slave flip-flop. This type of flip-flop (with some modification) is the basic building block used in most counters and registers. In many of these applications, it is desirable to have the flip-flop change states each time a true state occurs at its input. This is not possible with the master-slave flip-flop just described since it can only be set by data entered at the data set input and can only be cleared by data entered at the data reset input. If the flip-flop could be made to sense its present state and to direct any input information only to the appropriate input of the master flip-flop to change the state of the flip-flop, then the above requirement would be met.

In practice this is accomplished by: (a) feeding back output information (which represents the current state of the flip-flop) to the third inputs of gates G1 and G2; and (b) tying the two data inputs together.

The figure shown below illustrates the appropriate connections. (Note only the pertinent gates are shown.) The resulting flip-flop is called a J-K flip-flop. (Note that a JK flip-flop need not be in a master/slave configuration; a simple clocked RS FF could be modified at the input gates to be a JK FF. Also, the J and K inputs can be used independently and need not be tied together except when the flip-flop is being used as a data latch as below, e.g., compare the 7473 J-K flip-flop chip.)



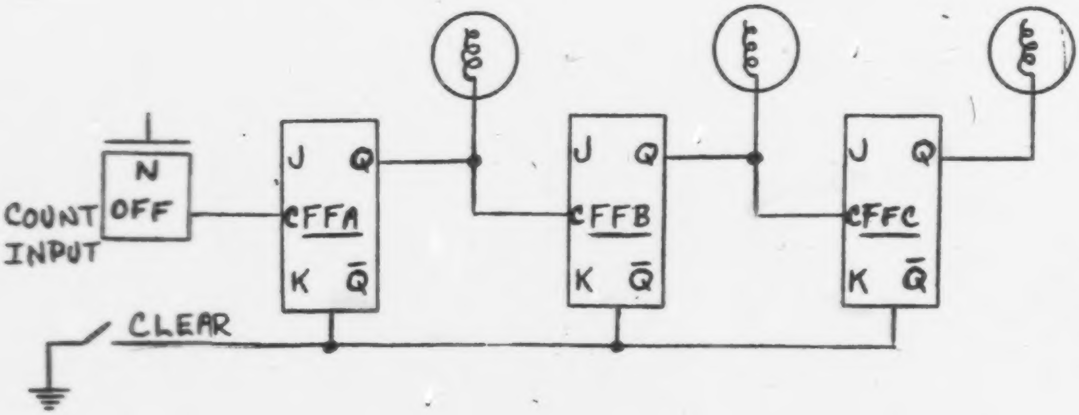
J-K FF (Master/Slave Configuration)

If time permits, make the suitable corrections to convert your master-slave flip-flop into a J-K flip-flop and perform the following sequence of experiments. With the data input disabled, enable and disable the clock input several times. Any change? Now enable the data input and then enable the clock input several times, noting the effect of each clock pulse on the output. Does the inclusion of the J-K connections affect the direct set-reset operations? This ability to prevent or permit the passage of clock pulses into a flip-flop (and therefore counters) is very important in the construction of counters with special count sequences as well as in control functions.

Counters.

As has already been mentioned, one of the primary uses for flip-flops is in the construction of various types of counters and registers. For, just as flip-flops consist of appropriately connected NAND gates, counters consist of appropriately wired flip-flops. The most commonly used flip-flop in modern integrated circuit counters is the J-K flip-flop that was discussed above. It can be used to construct two basic types of counters, asynchronous and synchronous, which can count up or down in a variety of counting schemes.

The simplest type of counter to be constructed is the asynchronous binary up-counter shown below:



Asynchronous Binary Counter

Wire this circuit using the SN7473 J-K flip-flop chips provided, and use the direct clear input to set the counter to an initial count of zero. Enable the count input several times and enter the result (indicator states) for each pulse in the table below:

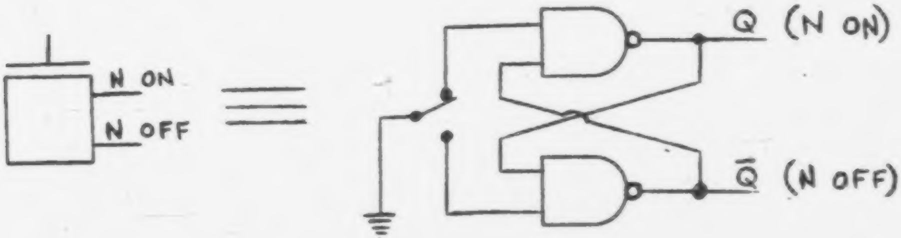
COUNT NO.	FFC(2^2)	FFB(2^1)	FFA(2^0)
0			
1			
2			
3			
4			
5			
6			
7			
8			

The completed table represents the binary counting sequence for a 3-bit counter.

Now connect the J input of flip-flop A to the $\overline{Q}$ output of flip-flop C. Clear all the flip-flops and note the effect of this change on the counting sequence. Explain.

This experiment demonstrates how the state of one flip-flop can be used to control the level input of another flip-flop. The same design can be applied to a number of other simple counters. For example, how would you construct a binary down-counter using these flip-flops? And how would you construct a gated up-down counter?

As an additional exercise, substitute a toggle switch (normally off) for the pulser that has been used up to now to provide the count input. Does the counter still function reliably? Probably not. The count should be inconsistent and unpredictable even though the circuit is wired precisely as before. The explanation resides in the fundamental differences between the two types of switches that are being used. The push-button pulsers are really RS flip-flops with a SPDT push-button switch which can apply a ground input to either the R or S inputs as shown below:



Thus, the type of switch provided by the pulser is often referred to as a "buffered" switch, because it switches output states smoothly at the first instant that the mechanical switch inputs a ground signal to the R or S input--and remains in the new state even if the mechanical switch should "bounce". The toggle switches, on the other hand, provide mechanical switching only. The resulting transitions have associated with them a certain amount of "bounce", oscillation or "ringing" which can often be of sufficient magnitude to repetitively trigger a counter. Thus, the use of "unbuffered" toggle switches may well lead to erroneous and unpredictable counting.

The 3-bit counter just examined can be used to count by five rather than by eight, provided that appropriate gating is used. The object is to have the counter clear itself on every fifth count. Consider the count sequence in the following table:

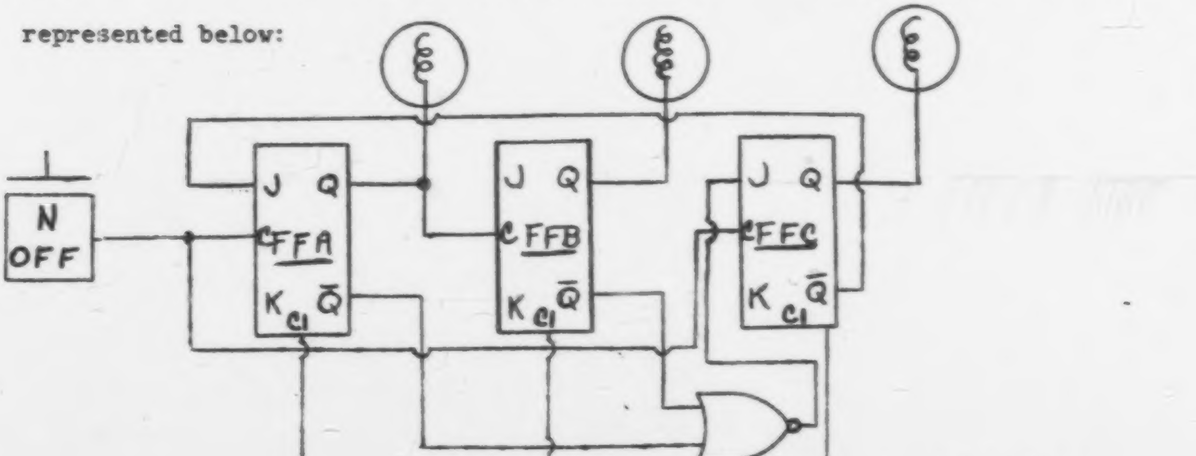
	COUNT NO.	FFC	FFB	FFA
	0	0	0	0
	1	0	0	1
	2	0	1	0
	3	0	1	1
NORMAL BINARY SEQUENCE	4	1	0	0
	5	1	0	1
DESIRED SEQUENCE	5	0	0	0

The normal count sequence and the desired sequence are represented. It will be observed that the desired sequence can be achieved if:

a) The fifth pulse is prevented from setting flip-flop A; and

b) The fifth pulse is used to clear flip-flop C.

Further examination of the table shows that the unique state of both flip-flops A and B being true can be used to control the desired count sequence. The count-of-five device can be implemented by: (a) feeding the count input directly to flip-flop C; (b) using the $\overline{Q}$ outputs of flip-flops A and B to enable flip-flop C when (and only when) the first two flip-flops are true; and (c) using the $\overline{Q}$ output of flip-flop C to disable the J input of flip-flop A when (and only when) C is true. The resulting circuit is represented below:

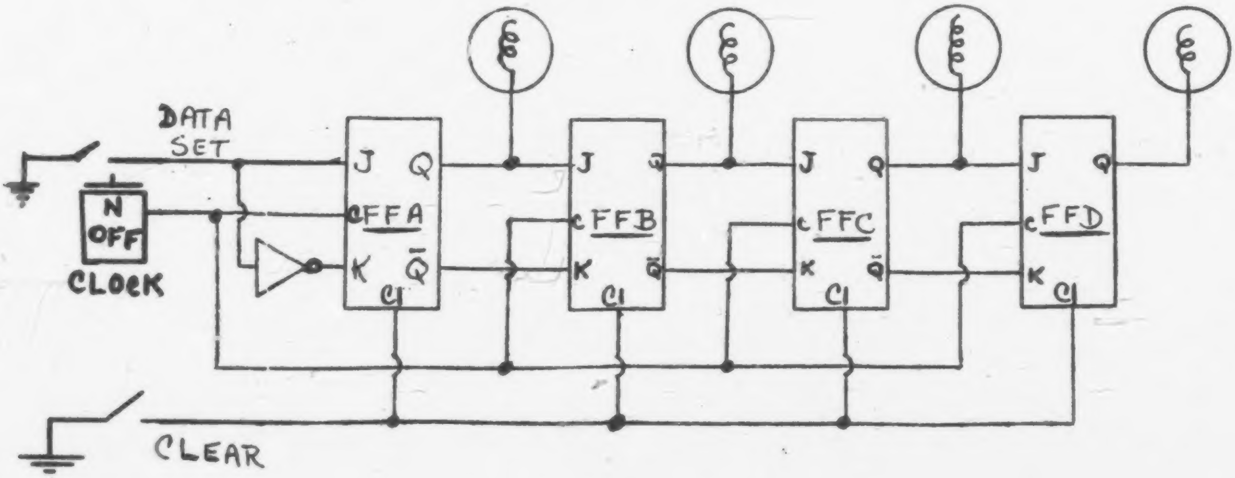


Construct this circuit and evaluate its operation to show that it does indeed count as predicted.

The count-of-five device can be converted to a count-of-ten device by inserting a single FF ahead of it. Use another flip-flop to construct a binary stage identical to one of the flip-flops employed in the binary counter. Connect the count input to the input of this binary stage and connect its Q output to the input of the count-of-five stage. Test the counter for counts up to ten and construct a table. It will be observed that if the four flip-flops are assigned the weights of 1(2^0), 2(2^1), 4(2^2), and 8(2^3), then each decimal number can be built up by a sum of these weighted values. The counter is called a Binary Coded Decimal (BCD) counter. This type of counter is very useful for decimal display of digital data. But in order to be useful in this fashion, the BCD output must be decoded to represent each decimal number.

Shift Registers.

A 4-bit *shift register* can be constructed from four J-K flip-flops (using 2 SN7473's) as shown below:



Predict and test the behavior of this circuit for the following conditions:

A. Initially clear all flip-flops and hold the Data Set input at logical 1. What will be outputs of the four FF's after four sequential clock pulses? Test this prediction.

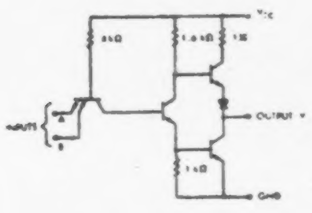
B. Repeat A, but this time connect the Data Set input to the $\overline{Q}$ output of the first FF. Explain the results.

C. Repeat A, but connect $\overline{Q}_D$ to the Data Set input. After initially clearing all of the FF's, allow ten or more clock pulses to be input, and observe the pattern generated by the indicator lamps.

Note that master-slave flip-flops were used to construct the shift register. Can you explain why a shift register would not work properly if constructed from simple clocked RS flip-flops?

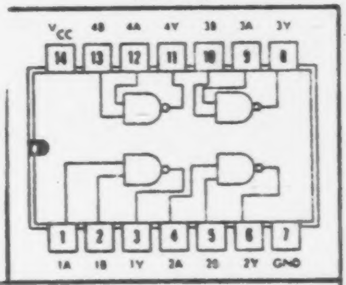
CIRCUIT TYPES . SN7400
QUADRUPLE 2-INPUT POSITIVE NAND GATES

schematic (each gate)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: $Y = \overline{AB}$

*recommended operating conditions

	MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}: SN5400 Circuits	4.5	5	5.5	V
SN7400 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N			10	
Operating Free-Air Temperature Range, T_A: SN5400 Circuits	−55	25	125	°C
SN7400 Circuits	0	25	70	°C

electrical characteristics over recommended operating free-air temperature (unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at both input terminals to ensure logical 0 level at output	1			2			V
$V_{in(0)}$	Logical 0 input voltage required at either input terminal to ensure logical 1 level at output	2					0.8	V
$V_{out(1)}$	Logical 1 output voltage	2	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	1	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	3	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	4	V_{CC} = MAX, V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current§	5	V_{CC} = MAX	SN5400	−20		−55	mA
				SN7400	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	6	V_{CC} = MAX, V_{in} = 5 V			12	22	mA
$I_{CC(1)}$	Logical 1 level supply current	6	V_{CC} = MAX, V_{in} = 0			4	8	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF, R_L = 400 Ω		7	15	ns
t_{pd1}	Propagation delay time to logical 1 level	66	C_L = 15 pF, R_L = 400 Ω		11	22	ns

---

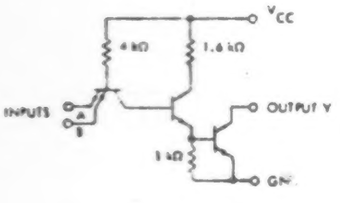
† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.

§ Not more than one output should be shorted at a time.

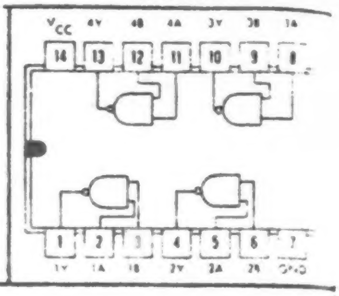
CIRCUIT TYPES SN7401
QUADRUPLE 2-INPUT POSITIVE NAND GATES
(WITH OPEN-COLLECTOR OUTPUT)

schematic (each gate)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: Y = $\overline{AB}$

recommended operating conditions

	MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}: SN5401 Circuits	4.5	5	5.5	V
SN7401 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N			10	
Operating Free-Air Temperature Range, T_A: SN5401 Circuits	−55	25	125	°C
SN7401 Circuits	0	25	70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at both input terminals to ensure logical 0 (on) level at output	1			2			V
$V_{in(0)}$	Logical 0 input voltage required at either input terminal to ensure logical 1 (off) level at output	7					0.8	V
$I_{out(1)}$	Output reverse current	7	V_{CC} = MIN, $V_{out(1)}$ = 5.5 V	V_{in} = 0.8 V,			250	µA
$V_{out(0)}$	Logical 0 output voltage (on level)	1	V_{CC} = MIN, I_{sink} = 16 mA	V_{in} = 2 V,			0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	3	V_{CC} = MAX,	V_{in} = 0.4 V			−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	4	V_{CC} = MAX,	V_{in} = 2.4 V			40	µA
			V_{CC} = MAX,	V_{in} = 5.5 V			1	mA
$I_{CC(0)}$	Logical 0 level supply current	6	V_{CC} = MAX,	V_{in} = 5 V		12	22	mA
$I_{CC(1)}$	Logical 1 level supply current	6	V_{CC} = MAX,	V_{in} = 0		4	8	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER		TEST FIGURE	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF,	R_L = 400 Ω		8	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF,	R_L = 4 kΩ		35	45	ns

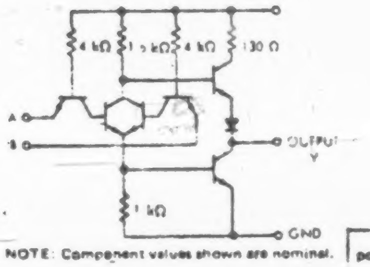
---

†For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡All typical values are at V_{CC} = 5 V, T_A = 25°C.

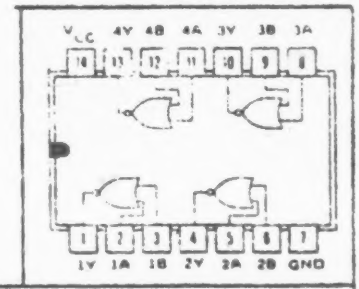
CIRCUIT TYPES SN7402
QUADRUPLE 2-INPUT POSITIVE NOR GATES

schematic (each gate)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: $Y = \overline{A + B}$

recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5402 Circuits	4.5	5	5.5	V
	SN7402 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5402 Circuits	−55	25	125	°C
	SN7402 Circuits	0	25	70	°C

electrical characteristics (over recommended operating free-air temperature range unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at either input terminal to ensure logical 0 level at output	8			2			V
$V_{in(0)}$	Logical 0 input voltage required at both input terminals to ensure logical 1 level at output	9					0.8	V
$V_{out(1)}$	Logical 1 output voltage	9	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	10	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	11	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	12	V_{CC} = MAX, V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current§	13	V_{CC} = MAX	SN5402	−20		−55	mA
				SN7402	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	14	V_{CC} = MAX, V_{in} = 5 V			14	27	mA
$I_{CC(1)}$	Logical 1 level supply current	14	V_{CC} = MAX, V_{in} = 0			8	16	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF, R_L = 400 Ω		8	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF, R_L = 400 Ω		12	22	ns

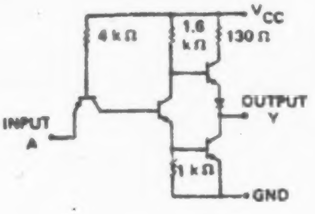
† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.

§ Not more than one output should be shorted at a time.

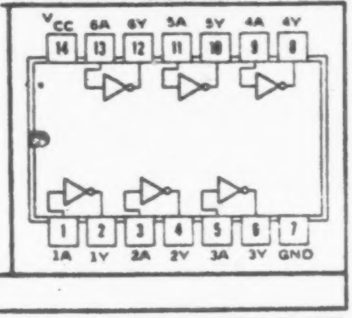
CIRCUIT TYPES SN7404
HEX INVERTERS

schematic (each inverter)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: $Y = \bar{A}$

recommended operating conditions

	MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}: SN5404 Circuits	4.5	5	5.5	V
SN7404 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N			10	
Operating Free-Air Temperature Range, T_A: SN5404 Circuits	−55	25	125	°C
SN7404 Circuits	0	25	70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at input terminal to ensure logical 0 level at output	15			2			V
$V_{in(0)}$	Logical 0 input voltage required at any input terminal to ensure logical 1 level at output	16					0.8	V
$V_{out(1)}$	Logical 1 output voltage	16	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 µA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	15	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current	18	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current	18	V_{CC} = MAX, V_{in} = 2.4 V				40	µA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current§	19	V_{CC} = MAX	SN5404	−20		−55	mA
				SN7404	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	20	V_{CC} = MAX, V_{in} = 5 V			18	33	mA
$I_{CC(1)}$	Logical 1 level supply current	20	V_{CC} = MAX, V_{in} = 0			6	12	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF,	R_L = 400 Ω		8	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF,	R_L = 400 Ω		12	22	ns

---

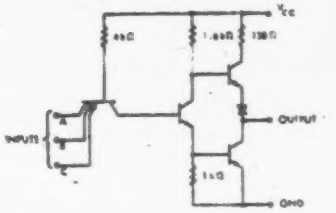
† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.

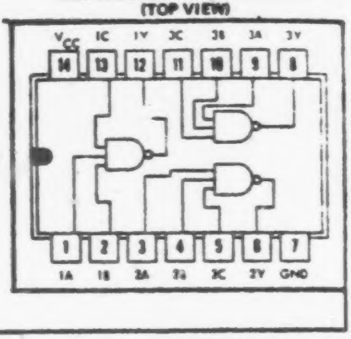
§ Not more than one output should be shorted at a time.

CIRCUIT TYPES , SN7410
TRIPLE 3-INPUT POSITIVE NAND GATES

schematic (each gate)



J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



NOTE: Component values shown are nominal.

positive logic: $Y = \overline{ABC}$

recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5410 Circuits	4.5	5	5.5	V
	SN7410 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5410 Circuits	−55	25	125	°C
	SN7410 Circuits	0	25	70	°C

electrical characteristics (over recommended operating free-air temperature range unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at all input terminals to ensure logical 0 level at output	1			2			V
$V_{in(0)}$	Logical 0 input voltage required at any input terminal to ensure logical 1 level at output	2					0.8	V
$V_{out(1)}$	Logical 1 output voltage	2	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	1	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	3	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	4	V_{CC} = MAX, V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short circuit output current§	5	V_{CC} = 5.5 V	SN5410	−20		−55	mA
				SN7410	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	6	V_{CC} = MAX V_{in} = 5 V			9	16.5	mA
$I_{CC(1)}$	Logical 1 level supply current	6	V_{CC} = MAX, V_{in} = 0			3	6	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF, R_L = 400 Ω		7	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF, R_L = 400 Ω		11	22	ns

---

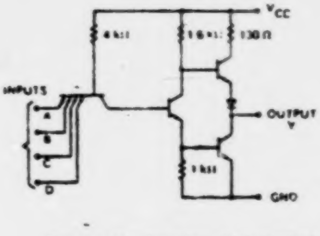
† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.

§ Not more than one output should be shorted at a time.

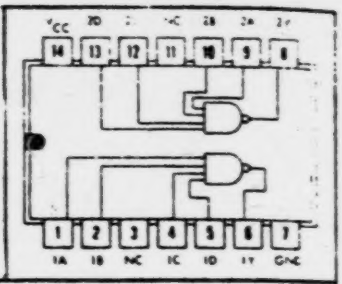
CIRCUIT TYPES SN7420
DUAL 4-INPUT POSITIVE NAND GATES

schematic (each gate)



Component values shown are nominal.
NC—No Internal Connection

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: $Y = \overline{ABCD}$

recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5420 Circuits	4.5	5	5.5	V
	SN7420 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5420 Circuits	−55	25	125	°C
	SN7420 Circuits	0	25	70	°C

electrical characteristics (over recommended operating free-air temperature range unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at all input terminals to ensure logical 0 level at output	1			2			V
$V_{in(0)}$	Logical 0 input voltage required at any input terminal to ensure logical 1 level at output	2					0.8	V
$V_{out(1)}$	Logical 1 output voltage	2	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	1	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	3	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	4	V_{CC} = MAX, V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current§	5	V_{CC} = MAX	SN5420	−20		−55	mA
				SN7420	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	6	V_{CC} = MAX, V_{in} = 5 V			8	11	mA
$I_{CC(1)}$	Logical 1 level supply current	6	V_{CC} = MAX, V_{in} = 0			2	4	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF, R_L = 400 Ω		8	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF, R_L = 400 Ω		12	22	ns

---

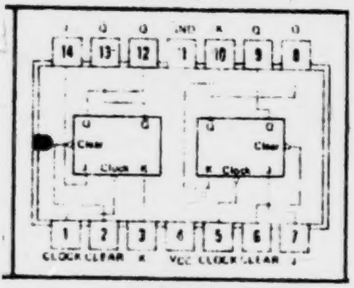
† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.

§ Not more than one output should be shorted at a time.

CIRCUIT TYPES , SN7473,
DUAL J-K MASTER-SLAVE FLIPS-FLOPS

SN5473, SN7473
J OR N DUAL-IN-LINE PACKAGE
(TOP VIEW)



positive logic:
Low input to clear sets Q to logical 0.
Clear is independent of clock.

description

These J-K flip-flops are based on the master-slave principle. Inputs to the master section are controlled by the clock pulse. The clock pulse also regulates the state of the coupling transistors which connect the master and slave sections. The sequence of operation is as follows: (See waveform on page 2-26)

1. Isolate slave from master
2. Enter information from J and K inputs to master
3. Disable J and K inputs
4. Transfer information from master to slave.

logic

TRUTH TABLE (Each Flip Flop)		
t_n		t_{n+1}
J	K	Q
0	0	Q_n
0	1	0
1	0	1
1	1	$\bar{Q}_n$

NOTES: 1. t_n = Bit time before clock pulse.
2. t_{n+1} = Bit time after clock pulse.

recommended operating conditions

	SN5473 SN54107			SN7473 SN74107			UNIT
	MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage V_{CC}	4.5	5	5.5	4.75	5	5.25	V
Normalized fan-out from each output, N			10			10	
Width of clock pulse, $t_{p(clock)}$ (See Figure 69)	20			20			ns
Width of clear pulse, $t_{p(clear)}$ (See Figure 70)	25			25			ns
Input setup time, t_{setup} (See Figure 69)	$\geq t_{p(clock)}$			$\geq t_{p(clock)}$			
Input hold time, t_{hold}	0			0			
Operating free-air temperature range, T_A	−55	25	125	0	25	70	°C

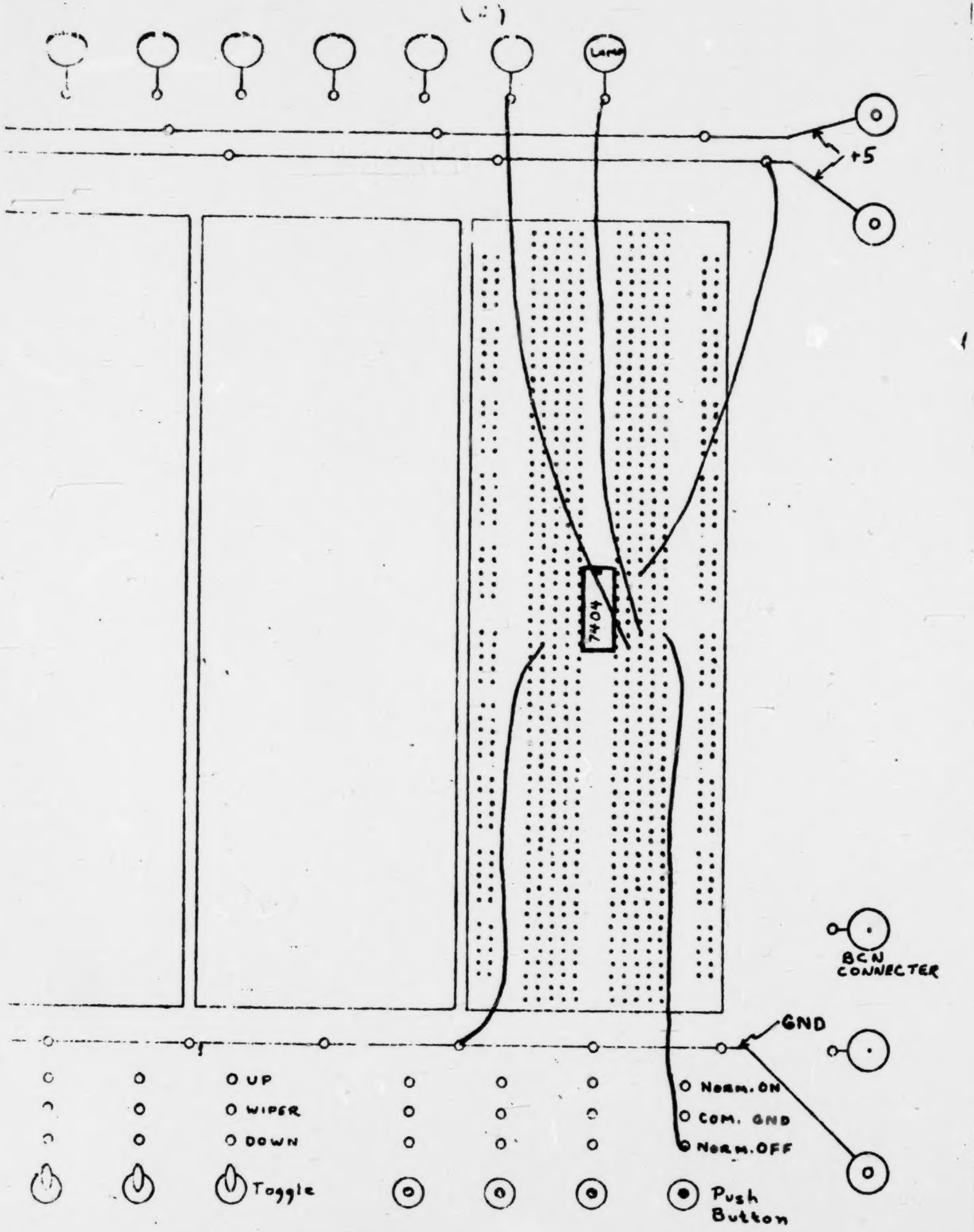
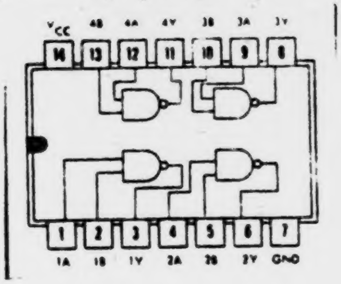
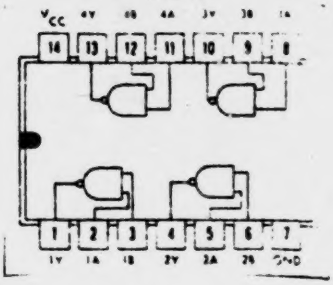

Lamp
+5
7404
BCN CONNECTER
GND
UP
WIPER
DOWN
Toggle
Norm. ON
COM. GND
Norm. OFF
Push Button

Fig. 1. ELITE LOGIC LAB

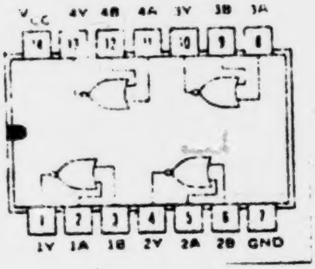
7400


VCC 4B 4A 4Y 3B 3A 3Y
14 13 12 11 10 9 8
1 2 3 4 5 6 7
1A 1B 1Y 2A 2B 2Y GND

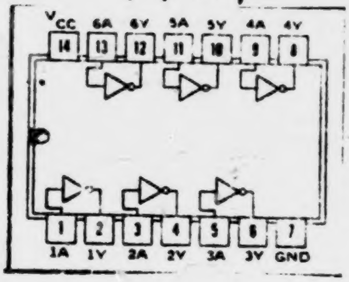
7401


VCC 4Y 4B 4A 3Y 3B 3A
14 13 12 11 10 9 8
1 2 3 4 5 6 7
1Y 1A 1B 2Y 2A 2B GND

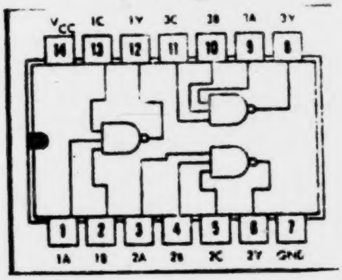
7402


VCC 4Y 4B 4A 3Y 3B 3A
14 13 12 11 10 9 8
1 2 3 4 5 6 7
1Y 1A 1B 2Y 2A 2B GND

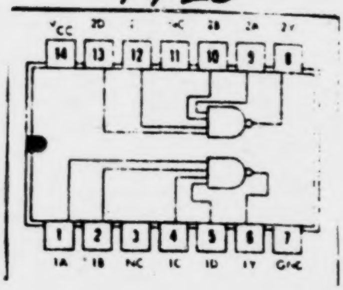
7404


VCC 6A 6Y 5A 5Y 4A 4Y
14 13 12 11 10 9 8
1 2 3 4 5 6 7
1A 1Y 2A 2Y 3A 3Y GND

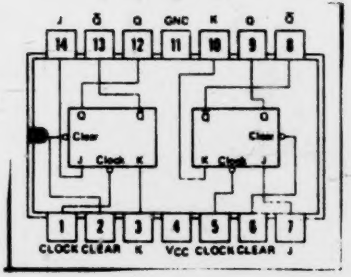
7410


VCC 1C 1Y 3C 3B 3A 3Y
14 13 12 11 10 9 8
1 2 3 4 5 6 7
1A 1B 2A 2B 3C 2Y GND

7420


VCC 2D NC 2B 2A 2Y
14 13 12 11 10 9 8
1 2 3 4 5 6 7
1A 1B NC 1C 1D 1Y GND

7473


J Q̄ Q GND K Q Q̄
14 13 12 11 10 9 8
Q Q̄ Clear J Clock K
Q Q̄ Clear K Clock J
1 2 3 4 5 6 7
CLOCK CLEAR K VCC CLOCK CLEAR J

LOGIC EXPERIMENT II

TIMING AND SYNCHRONIZATION

Purpose.

The purpose of this experiment is to introduce the concepts of timing and synchronization of events in digital circuitry. These topics are of primary importance in all types of digital circuitry whether it be a self-contained dedicated system or a general purpose computer interface. In most (if not all) digital systems, an error in timing of one or more events can invalidate the entire operation of the system.

Most digital systems contain a central clock (oscillator) or pulse generator which generates an uninterrupted pulse train with a definite time interval between pulses. This clock generates the time base for the system. The proper operation of a system may depend upon the synchronization of two or more events. For example, if it is desired that the clock time be related to the real elapsed time in an experiment then it is necessary that the initiation of the experiment be synchronized with the occurrence of a clock pulse which is then used as a reference point from which other time measurements for that experiment are made. The operation of a digital circuit, whether it be a single component of a larger system or an entire system, must be controlled. In other words, the occurrence of events must be properly sequenced to perform the desired function. This experiment is designed to demonstrate some of the circuits involved in these operations.

Revised April, 1973

Components.

Elite 3-a logic lab.

Dual trace oscilloscope.

Pulse generator, Model PG-1.

TTL logic chips:

SN7400	Quad 2-input NAND
SN74121	One shot (monostable multivibrator)
SN7476	Dual JK flip-flop
SN7404	Hex inverter
SN7442	Decoder (BCD → Decimal)
SN7495	4-bit shift register (data latch)
SN7410	Triple 3-input NAND
SN7420	Dual 4-input NAND
SN7402	Quad 2-input NOR

25 KΩ potentiometer

0.1 μf capacitor

Required Sections.

Component familiarization, Fig. II-1, II-2, II-3

Time Interval measurement, Fig. II-5

Synchronization, Fig. II-7, II-8, II-9

Flip-flop Time Delay, Fig. II-11

Optional Sections.

Timing, Decoder Data Latch, Fig. II-4

Time Interval Generator, Fig. II-6

Asynchronous Level Change-Pulse Synchronization, Fig. II-10

Component Familiarization.

It is necessary to fully understand the components which will be used to be able to build more complex logic circuits. Here, three new components (the one shot, decoder, and shift register) will be introduced and some of the important characteristics of the components already used (pulse generator, gates, flip-flops, and counters) will be shown.

Oscilloscope. The oscilloscope provided should be a dual trace instrument capable of at least a 200 nsec. time scale sensitivity and internal or external triggering. Become familiar with the various controls on this device, asking the instructor for aid when necessary.

Pulse Generator. The Model PG-1 pulse generator provides the timing used on the Elite lab. Display a pulse train on the oscilloscope using the internal scope trigger and Channel A. Observe the pulse characteristics. Pulse height may be varied between 0 and +10 volts and the maximum pulse width is about 100msec. The maximum frequency is about 1 MHz. Observe the output pulses by setting all dials on the pulse generator to their maximum positions. Scope settings of about 5V/cm (vertical) and 0.5 μsec/cm (horizontal) should permit well-defined pulses to be displayed. Vary the pulse generator controls and note each result.

One-Shot - SN74121 - The function of the one-shot (O.S.) is to generate a controllable delay after being triggered. Connect an O.S. to the pulse generator and oscilloscope as shown in Figure II-1. In this mode, the O.S. will pulse continuously when SW1 is closed. Observe the effect of the potentiometer on the delay time. Notice that the one-shot gives a broad 0-to-1-to-0 transition. In other words, on the time scale of the P.G., these changes are more properly considered level changes than pulses. A .012 μF capacitor may be substituted for the 0.1 μF capacitor to obtain shorter time delays.

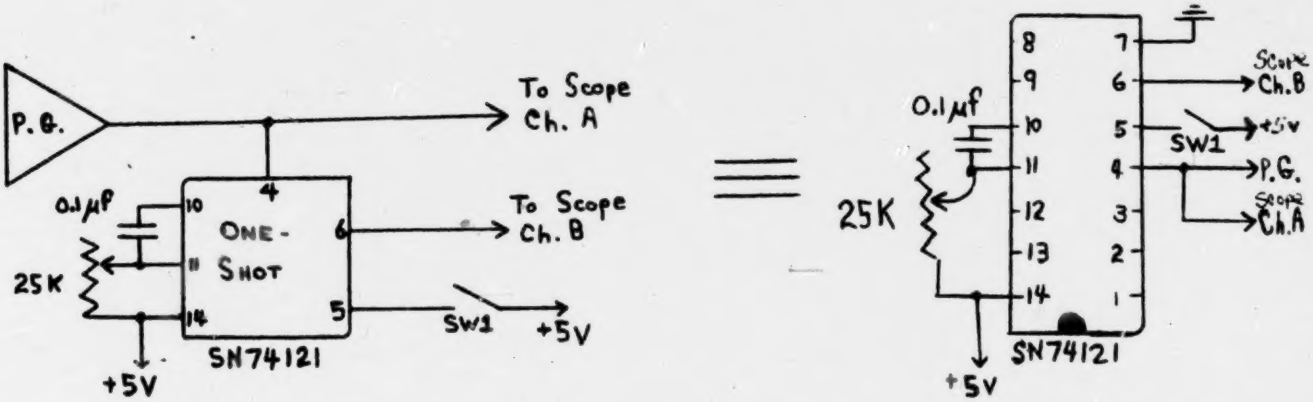

P.G.
To Scope Ch. A
To Scope Ch. B
0.1µf
25K
10
11
14
4
6
5
ONE-SHOT
SN74121
SW1
+5V
+5V
8
9
10
11
12
13
14
7
6
5
4
3
2
1
0.1µf
25K
Scope Ch.B
+5v
SW1
P.G.
Scope Ch.A
SN74121
+5V

Figure II-1 (Required)
One-Shot

Decoder - SN7442 - The BCD-to-decimal decoder may be tested as shown in Figure II-2. Wire the circuit as shown. Prepare a truth table for this device. (Power and ground connections for the chips will no longer be shown in the following figures.) Make sure of BCD Input logic levels.

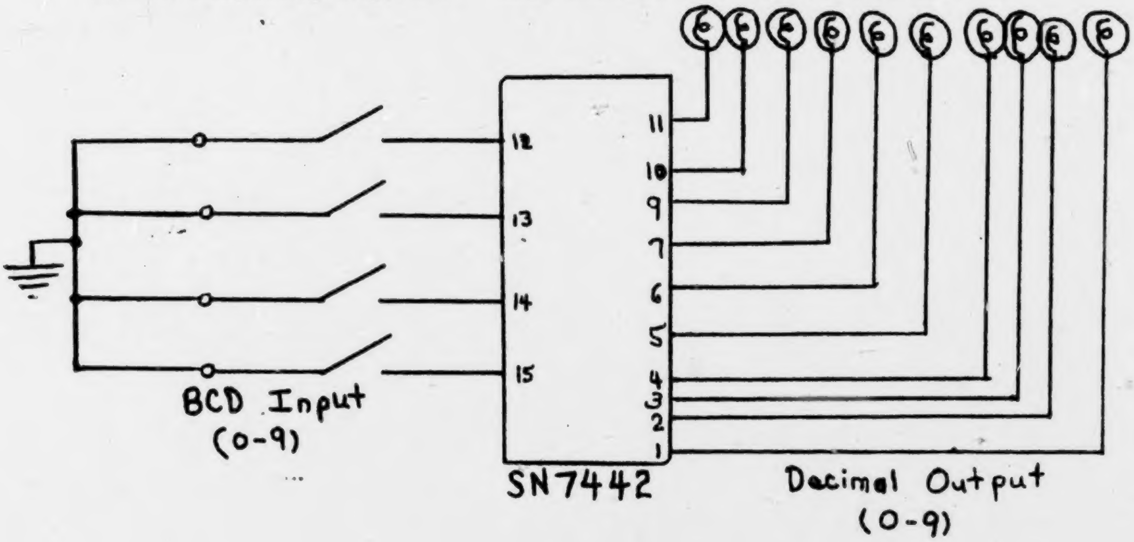

6
6
6
6
6
6
6
6
6
6
12
13
14
15
11
10
9
7
6
5
4
3
2
1
BCD Input
(0-9)
SN7442
Decimal Output
(0-9)

Figure II-2 (Required)
Decoder

Shift Register/Data Latch - SN7495 - In Logic Experiment I, a 4-bit serial shift register was built. The SN7495 is a single I.C. 4-bit shift register. However, this device is more versatile in that it can also function as a 4-bit data latch. Parallel or serial input or output are available. The data latch function is shown by implementing the circuit in Figure II-3. The external command is provided by a one-shot. The one-shot pulse must be long enough to allow the register to shift the input code from the input to the output. Vary the O.S. delay time and note its effect on the data transfer. A pulse of about 3-5 µsec should be necessary for data transfer to occur.

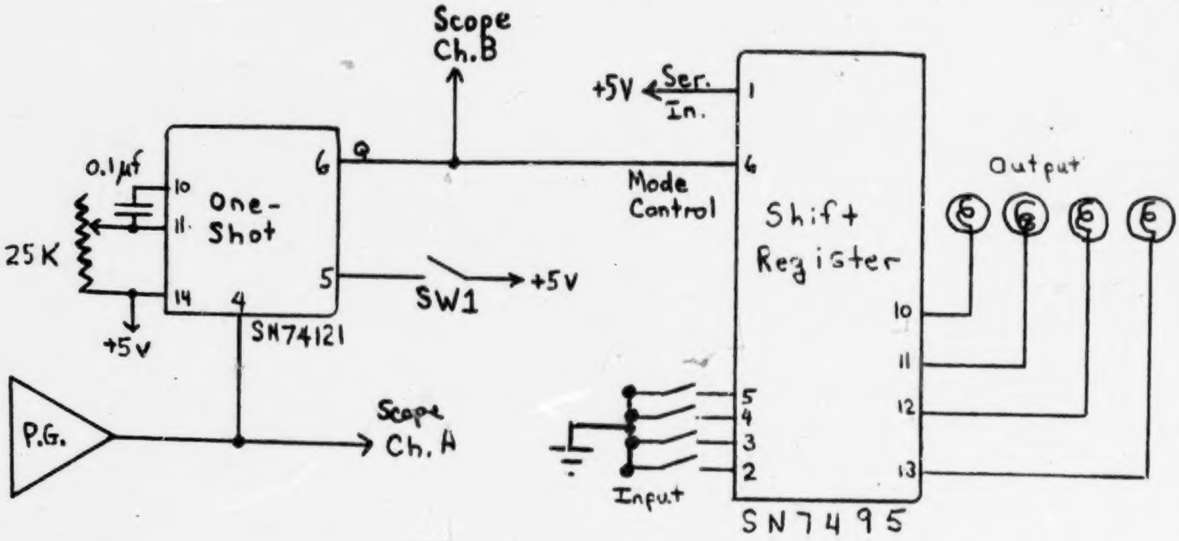

Scope Ch.B
+5V
Ser. In.
1
0.1µf
10
11
One-Shot
6
Q
Mode Control
4
Shift Register
Output
25K
14
4
5
SW1
+5V
+5V
SN74121
10
11
P.G.
Scope Ch. A
5
4
3
2
12
13
Input
SN7495

Figure II-3 (Required)
Data Latch

Timing.

(OPTIONAL) Decoder-Data Latch. A combination of all the circuits used to this point is shown in Figure II-4. Very often in a computer interface, it is necessary to obtain a piece of coded data, decode it to a desired numerical system, and transfer it to a place where it may be readily accessed for further use. Also, the timing of the data conversion and transfer may be critical to the experimental procedure. Timing errors may be serious; for example, if the strobe pulse to the data latch is not long enough, the input data will not appear reliably at the output.

Wire the circuit as shown in Figure II-4. By enabling SW1, the time when the data is transferred from the decoder to the latch may be controlled by the one-shot. This may be shown by the following procedure. With SW1 closed (+5V), adjust the time delay of the one-shot output pulse to approximately 200 nsec with the 25K potentiometer. Open SW1. Apply a suitable code to the decoder input (0100) and enable SW1. Does the correct code from the decoder appear at the output of the shift register? It should not, and the shift register should remain unchanged. The pulse time from the one-shot was not sufficiently long to allow the decoder to "settle" with the decoded result at its output for transfer into the shift register. Increase the one-shot pulse to 10 μsec, open SW1, input a different code into the decoder (1000), enable SW1 and note the result. This time the correct code should appear at the shift register output. Vary the O.S. delay and note the delay time where the pulse is just long enough for the proper result to occur. There should be a region where the shift register will change its value, but not to the correct value.

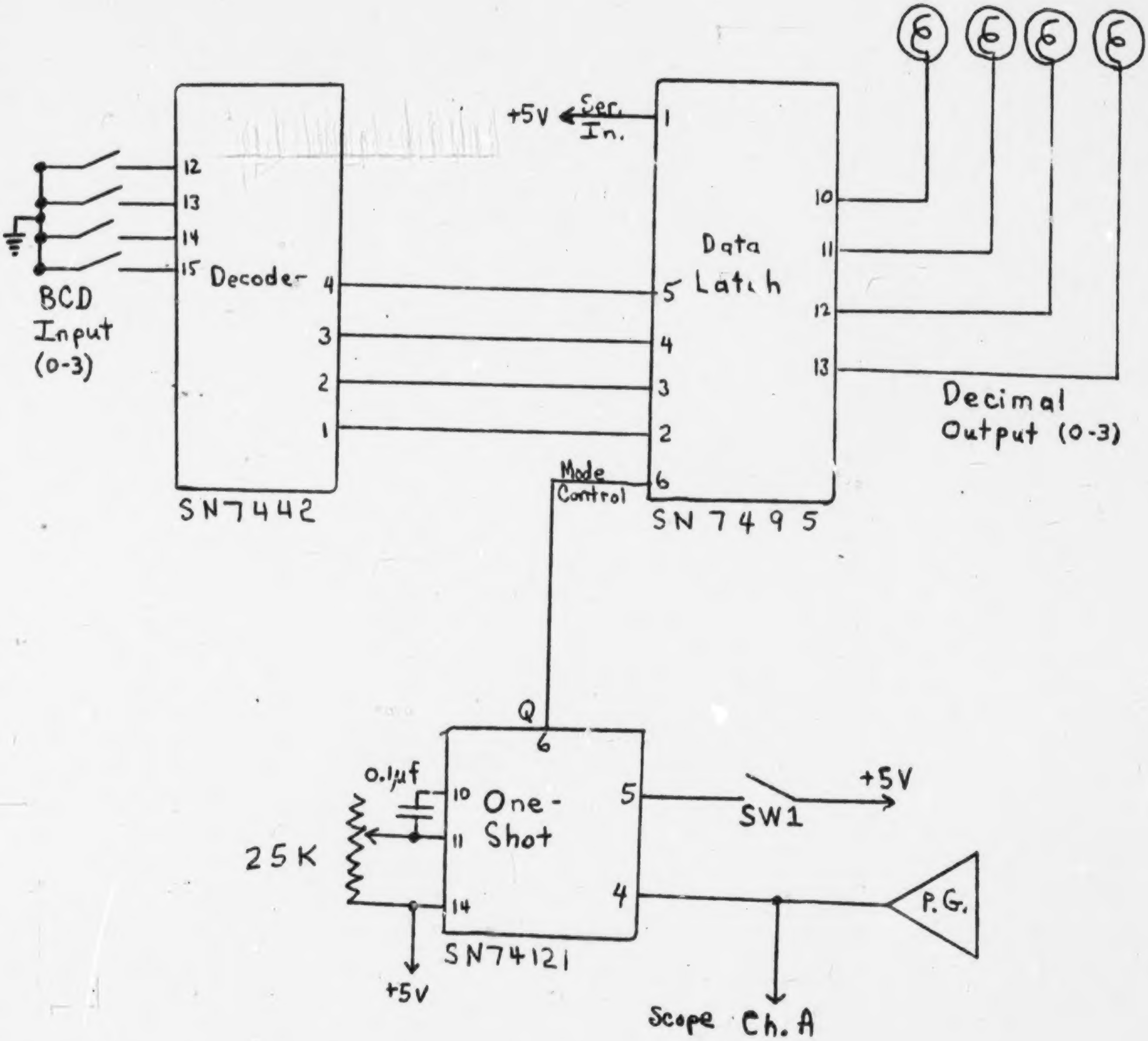

12
13
14
15
Decoder
BCD Input (0-3)
4
3
2
1
SN7442
+5V
Ser. In.
1
Data Latch
5
4
3
2
6
Mode Control
SN7495
10
11
12
13
Decimal Output (0-3)
Q
6
0.1μf
10
11
14
One-Shot
25K
5
4
SN74121
+5V
SW1
+5V
P.G.
Scope Ch. A

Figure II-4 (OPTIONAL)

Decoder-Data Latch

(Required) Time Interval Measurement. The time interval between pulses from separate pulse sources can be measured by making separate connections to the clock inputs of two flip-flops. The resulting circuit is shown in Figure II-5. A pulse (momentary closure of SW1) at the start input opens the counting gate and a pulse at the stop input (SW2) closes the counting gate. Accurately measure the pulse frequency of the pulse generator and adjust it to 10 Hz. All flip-flops must be cleared with P1 before any measurement can be made. The counter will measure the time between closing switches 1 and 2. If you are fast, you may need to use a higher clock frequency. Save this circuit if you plan to do the next circuit.

Time Interval Generator. (Optional) A precision time interval generator may be constructed with the modifications made to the previous circuit as shown in Figure II-6. The generated time interval begins when the counter begins counting clock pulses and stops when a predetermined number of clock pulses have been counted. The preset count is established by selecting the input levels to the presettable decoder.

Wire the circuit in Figure II-6. Preset the counter to 1111 rather than 0000 so that the first count produces a "0" level output at the zero decoder gate resulting in a "1" at the time interval output. Note that when the preset count is reached, the indicator lamp L1 goes off. The presettable decoder output applies a "0" level stop pulse to the count gate control and to the output bistable gate circuit terminating the output pulse. After the counter and count gate control circuits are reset, the circuit can again be triggered with SW1 to generate another precision duration pulse. Use a clock input of 10 Hz and generate pulses of 0.1, 0.5, 1.0, and 1.5 seconds duration. Observe the pulse duration with both an indicator light and the oscilloscope set on a very slow scan rate.

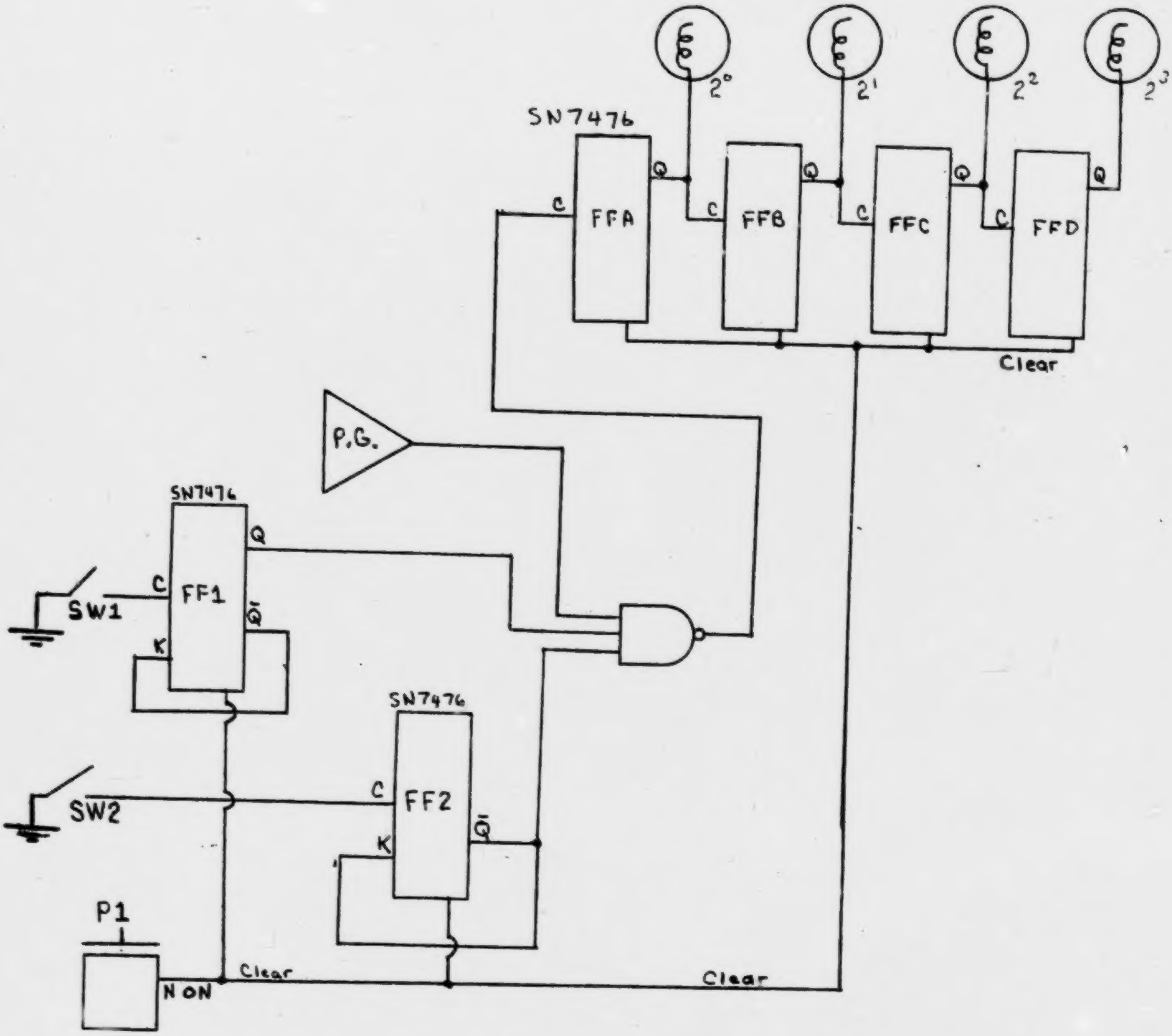

SN7476
FFA
FFB
FFC
FFD
C
Q
2^0
2^1
2^2
2^3
Clear
P.G.
SN7476
FF1
C
K
Q
$\bar{Q}$
SW1
SN7476
FF2
C
K
$\bar{Q}$
SW2
P1
N ON
Clear
Clear

Figure II-5 (Required)
Output Bistable Gate Circuit

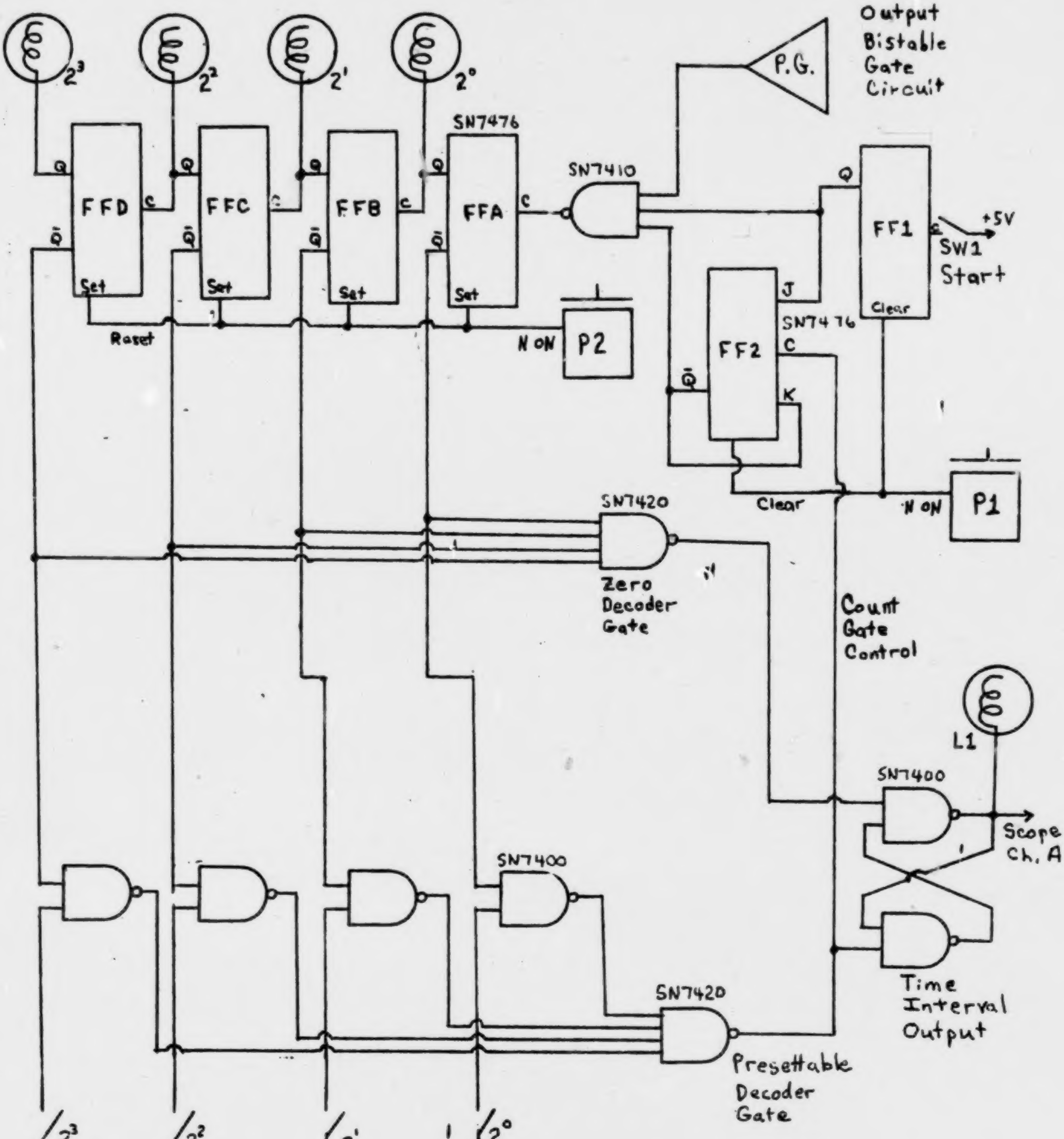

2³
2²
2¹
2⁰
Output Bistable Gate Circuit
P.G.
SN7476
FFD
FFC
FFB
FFA
Q
Q̄
C
Set
Reset
SN7410
FF1
+5V
SW1 Start
Clear
J
SN7476
FF2
C
K
Q̄
N ON
P2
Clear
N ON
P1
SN7420
Zero Decoder Gate
Count Gate Control
L1
SN7400
Scope Ch. A
Time Interval Output
SN7400
SN7420
Presettable Decoder Gate
Presettable Decoder Inputs

Figure II-6 (OPTIONAL)

Time Interval Generator

Synchronization.

Often it is necessary to synchronize randomly occurring (asynchronous) pulse or level changes with the system clock. Consider the following problem. Suppose it is necessary to output clock pulses on two lines labeled A and B. A pulse train is wanted on line A only if a controlling level is true, and a pulse train on line B is wanted only if the same controlling level is false. In addition, the first appearance of a pulse should be synchronized with the system clock. A first solution to be considered might be to use a pair of AND gates to pass the clock circuit enabling one gate (A) when the controlling level is true and enabling the other level when the controlling level is false. A typical circuit would be that in Figure II-7.

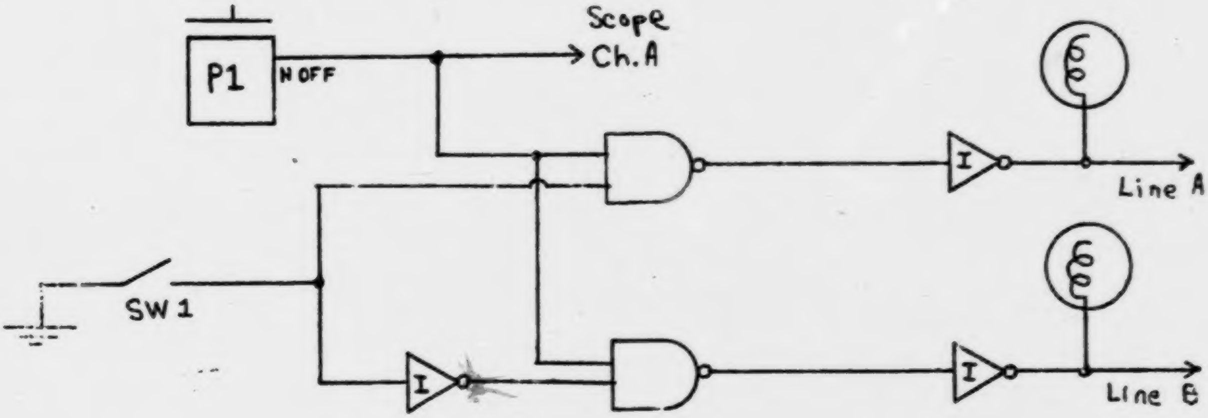


Figure II-7 (Required)
Pulse and Level Change Synchronization

Construct this circuit and test it as follows. Put the scope on a very slow scan rate (1 sec/div), open SW1 and press and release the pulser several times, noting the behavior of line A relative to the pulser output. (Use a 0-1-0 pulse (Norm OFF).) Now connect line B to channel B of the scope. Push the pulser button (logical 1) and, while holding it closed, close SW1. Then release the pulser switch. Is the change at line B synchronized with the change in pulser output? Does this circuit fulfill all the requirements set down above? The answer is no. This may be explained by considering the timing diagram in Figure II-8.

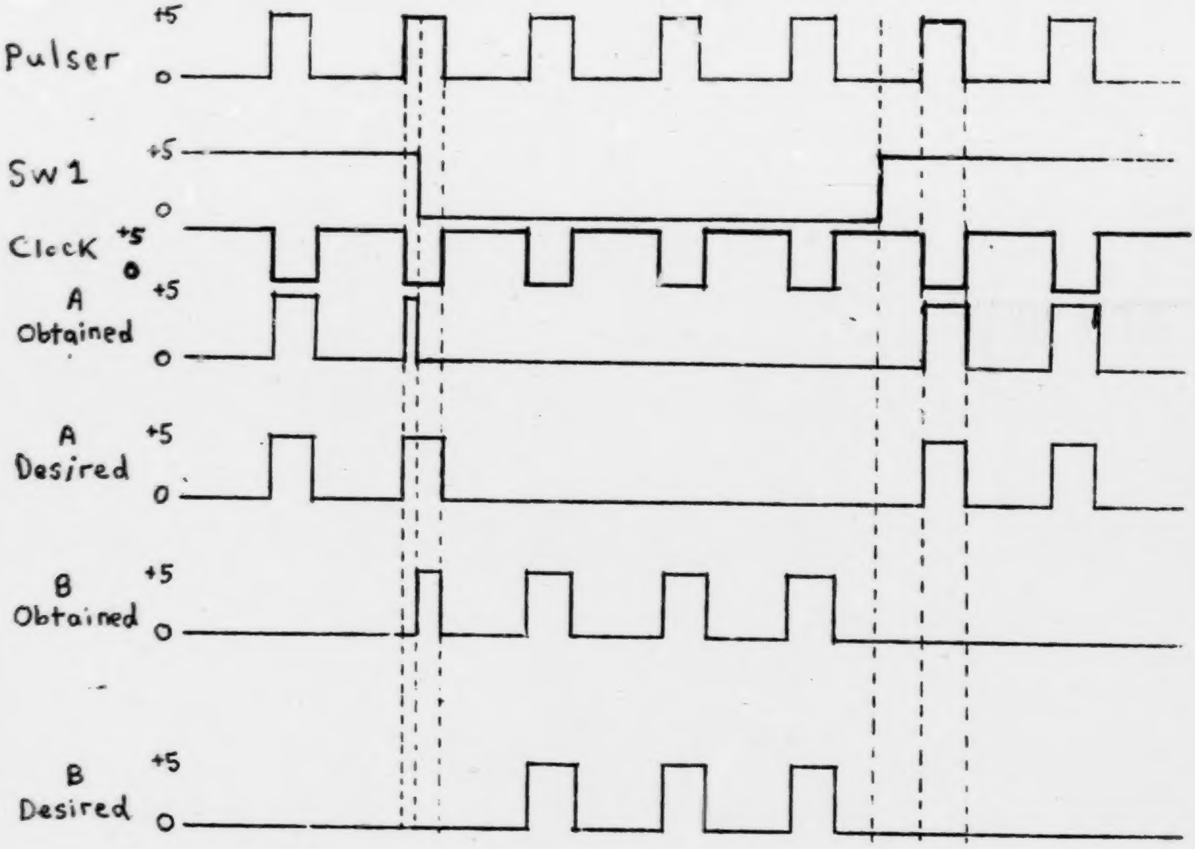

Pulser
+5
0
Sw 1
+5
0
Clock
+5
0
A Obtained
+5
0
A Desired
+5
0
B Obtained
+5
0
B Desired
+5
0

Figure II-8 (Required)
Timing Diagram

It can be seen that if the level change at SW1 occurs between pulses, the outputs on channels A and B are as desired; that is, only the first whole pulse after the SW1 level change is seen at Channel A when SW1 is logical 1 and at Channel B when SW1 is logical 0. However, when the SW1 level change occurs during a pulse, the levels at both A and B change immediately in the middle of a pulse. This produces a shortened pulse on both lines A and B, which could cause problems with any device which requires a specific pulse width. Moreover, the pulses occur at the wrong time. Synchronization of lines A and B with the pulser will be maintained if only the first complete pulse after the switch change appears at A and B.

Test this circuit further by applying a 1-0-1 pulse with the pulser (Norm ON) or by using the pulse generator through an inverter. Construct the timing diagrams for this situation and compare with Figure II-8. Note that the deviation from the desired output on Channels A and B occurs when the switch change occurs between pulses.

The circuit in Figure II-9, employing the memory capability of a JK master-slave flip-flop, resolves the problem seen above. Construct this circuit and test it as suggested for Figure II-7 above, except the pulser switch should be opened and closed at least twice after closing SW1. Use 0-1-0 pulses. The flip-flop triggers on a 1-0 level change. Replace the pulser in Figure II-9 with the pulse generator operating at a low frequency

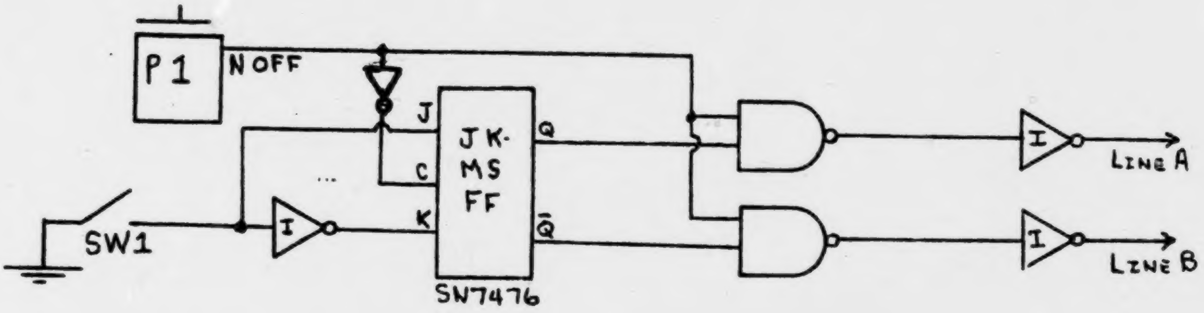


Figure II-9 (Required)
Flip-flop Synchronization

and note the effect of opening and closing SW1 on lines A and B relative to the P.G. Check to see if this circuit operates as desired using 1-0-1 pulses. Remember that the flip-flop is negative-edge triggered. Save this circuit if you plan to do the next section.

Asynchronous Level Change-Pulse Synchronization. (Optional) Consider the problem of converting an asynchronous level change or trigger pulse (0 to 1) to a single synchronous pulse corresponding exactly to a pulse supplied by the pulse generator. The circuit in Figure II-10 provides a solution to this problem. Construct and test this circuit. First, clear the FF's with P2 and confirm that the circuit performs the desired function. A single pulse should appear at the output for a single 1-0 change at P1. This pulse should be the same as a single pulse from the pulse generator. Construct timing diagrams for the three indicator lights in the space below:

Flip-Flops and Counters, Time Delays. In most applications, it is assumed that the output of a binary counter is synchronized with the input clock pulses. Flip-flops are made up of NAND gates and therefore have propagation delays associated with them. To illustrate this point, wire the

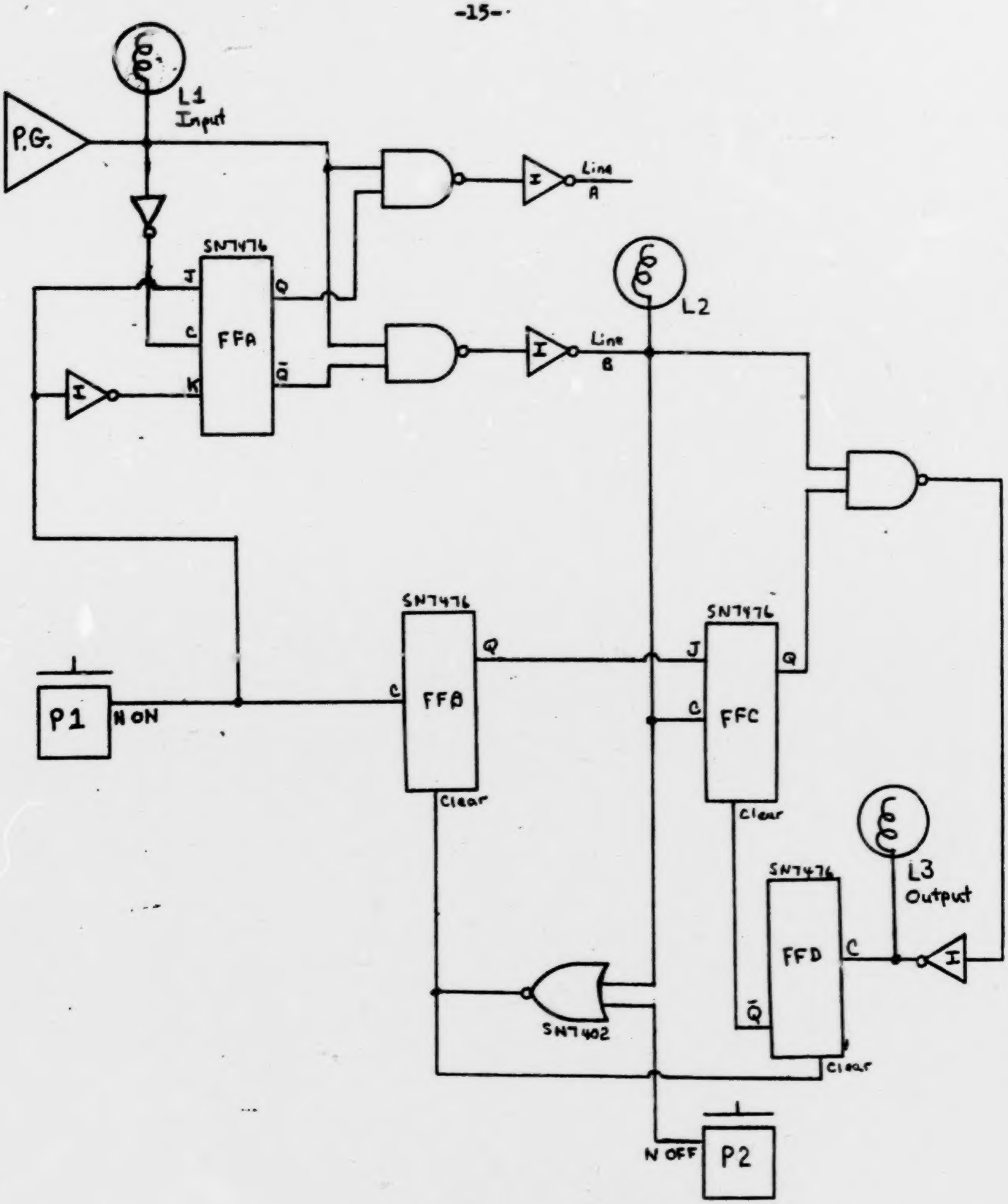

L1
Input
P.G.
Line A
SN7476
J
Q
C
FFA
K
Q̄
I
Line B
L2
SN7476
FFB
Q
C
Clear
P1
N ON
SN7476
FFC
J
C
Q
Clear
L3
Output
SN7476
FFD
C
Q̄
Clear
SN7402
N OFF
P2

Figure II-10 (OPTIONAL)

Asynchronous Level Change-Pulse Synchronization

circuit shown in Figure II-11, an asynchronous ripple-carry counter. Use about a 10 KHz pulse frequency and a 10 μsec pulse width from the pulse generator. Adjust the one-shot time delay to about 50 μsec and set the scope to a 200 nsec/div time scale. Observe the input pulse (Ch. A) and the pulse as it emerges from the counter into the O.S. (Ch. B). The one-shot delay and the P.G. frequency vernier will have to be varied to obtain a stable output on the scope. The circuit is designed to continually set and clear on alternate pulses allowing the desired signals to be studied continuously. Note that when the counter is set, the count "ripples" through all FF's. Note also that a full pulse will not be obtained at the final flip-flop Q output. A total time delay of about 300 nsec should be observed for the 10 FF's. This means that the time delay per flip-flop is only about 25-30 nsec. A 300 nsec delay can be a serious problem if extremely accurate timing is needed. Use of a parallel synchronous counter would reduce this delay significantly.

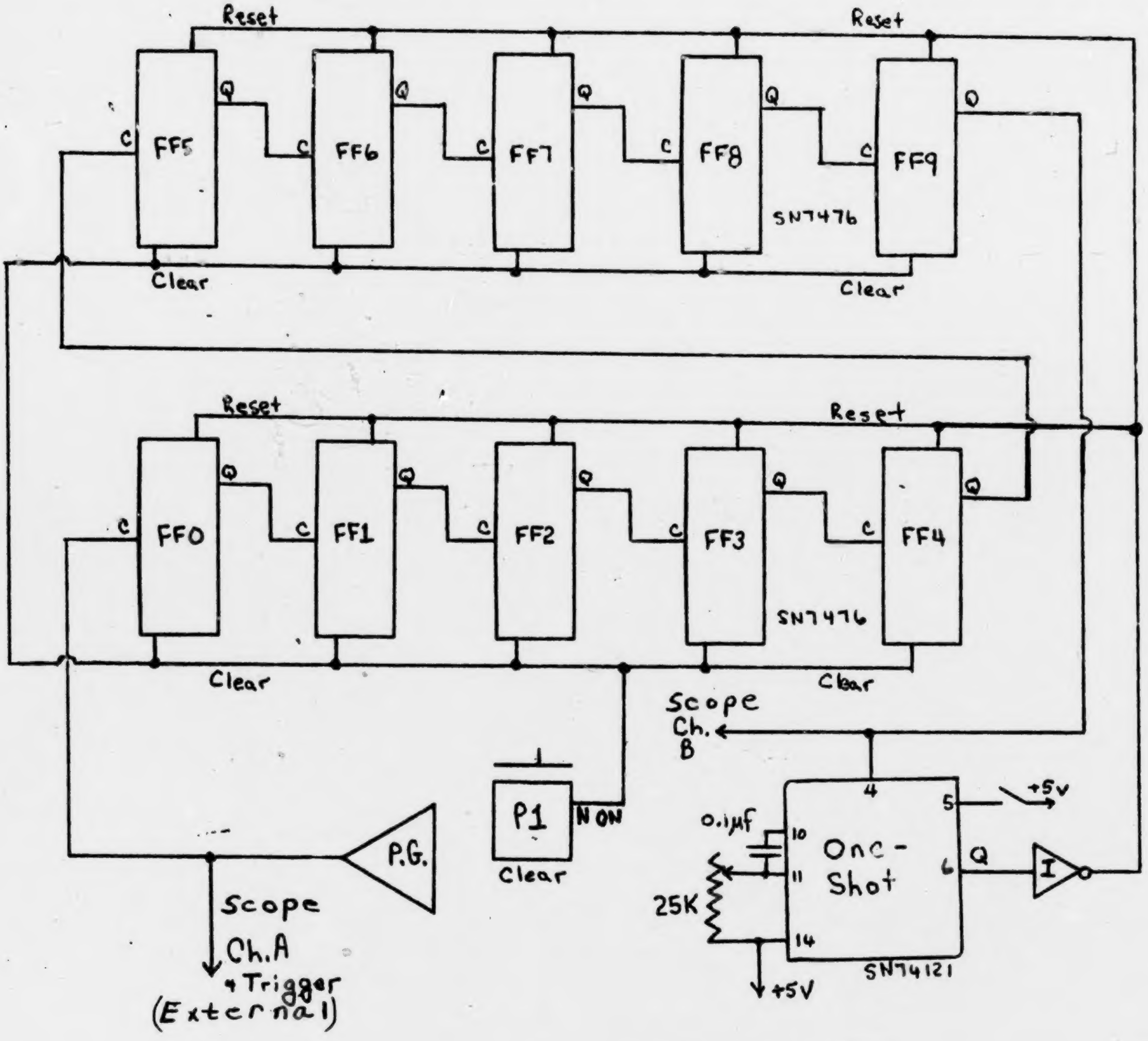

Reset
Reset
Q
Q
Q
Q
Q
C
C
C
C
C
FF5
FF6
FF7
FF8
FF9
SN7476
Clear
Clear
Reset
Reset
FF0
FF1
FF2
FF3
FF4
SN7476
Clear
Clear
Scope
Ch.
B
4
5
+5v
0.1μf
10
11
14
One-
Shot
6
Q
I
25K
+5V
SN74121
P1
N ON
Clear
P.G.
Scope
Ch.A
+Trigger
(External)

Figure II-11(Required)

Asynchronous Ripple-Carry Counter

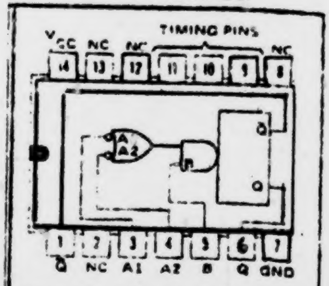
logic

TRUTH TABLE (See Notes 1 thru 3)

t_n INPUT			t_{n+1} INPUT			OUTPUT
A1	A2	B	A1	A2	B	
1	1	0	1	1	1	Inhibit
0	X	1	0	X	0	Inhibit
X	0	1	X	0	0	Inhibit
0	X	0	0	X	1	One Shot
X	0	0	X	0	1	One Shot
1	1	1	X	0	1	One Shot
1	1	1	0	X	1	One Shot
X	0	0	X	1	0	Inhibit
0	X	0	1	X	0	Inhibit
X	0	1	1	1	1	Inhibit
0	X	1	1	1	1	Inhibit
1	1	0	X	0	0	Inhibit
1	1	0	0	X	0	Inhibit

$1 = V_{in(1)} \geq 2\ V$
$0 = V_{in(0)} \leq 0.8\ V$

J OR N DUAL-IN-LINE PACKAGE
(TOP VIEW)
(See Notes 6 thru 9)



positive logic: see truth table and notes 5 and 6

NOTES:
1. t_n = time before input transition.
2. t_{n+1} = time after input transition.
3. X indicates that either a logical 0 or 1 may be present.
4. NC = No Internal Connection.
5. A1 and A2 are negative edge-triggered logic inputs, and will trigger the one shot when either or both go to logical 0 with B at logical 1.
6. B is a positive Schmitt trigger input for slow edges or level detection, and will trigger the one shot when B goes to logical 1 with either A1 or A2 at logical 0. (See Truth Table)
7. External timing capacitor may be connected between pin ⑩ (positive) and pin ⑪. With no external capacitance, an output pulse width of typically 30 ns is obtained.
8. To use the internal timing resistor (2 kΩ nominal), connect pin ⑨ to pin ⑭.
9. To obtain variable pulse width connect external variable resistance between pin ⑨ and pin ⑭. No external current limiting is needed.
10. For accurate repeatable pulse widths connect an external resistor between pin ⑪ and pin ⑭ with pin ⑨ open-circuit.

description

This monolithic TTL monostable multivibrator features d-c triggering from positive or gated negative-going inputs with inhibit facility. Both positive and negative-going output pulses are provided with full fan-out to 10 normalized loads.

Pulse triggering occurs at a particular voltage level and is not directly related to the transition time of the input pulse. Schmitt-trigger input circuitry (TTL compatible and featuring temperature-independent backlash, See Figure L) for the B input allows jitter-free triggering from inputs with transition times as slow as 1 volt/second, providing the circuit with an excellent noise immunity of typically 1.2 volts. A high immunity to V_{CC} noise of typically 1.5 volts is also provided by internal latching circuitry.

Once fired, the outputs are independent of further transitions on the inputs and are a function only of the timing components. Input pulses may be of any duration relative to the output pulse. Output pulse lengths may be varied from 40 nanoseconds to 40 seconds by choosing appropriate timing components. With no external timing components (i.e., pin ⑨ connected to pin ⑭, pins ⑩, ⑪ open) an output pulse of typically 30 nanoseconds is achieved which may be used as a d-c triggered reset signal. Output rise and fall times are TTL compatible and independent of pulse length.

Pulse width is achieved through internal compensation and is virtually independent of V_{CC} and temperature. In most applications, pulse stability will only be limited by the accuracy of external timing components.

CIRCUIT TYPES

SN7442,
4-LINE-TO-10-LINE DECODERS (1-OF-10)

- BCD-to-Decimal
- Excess-3-to-Decimal
- Excess-3 Gray-to-Decimal

Also for applications as

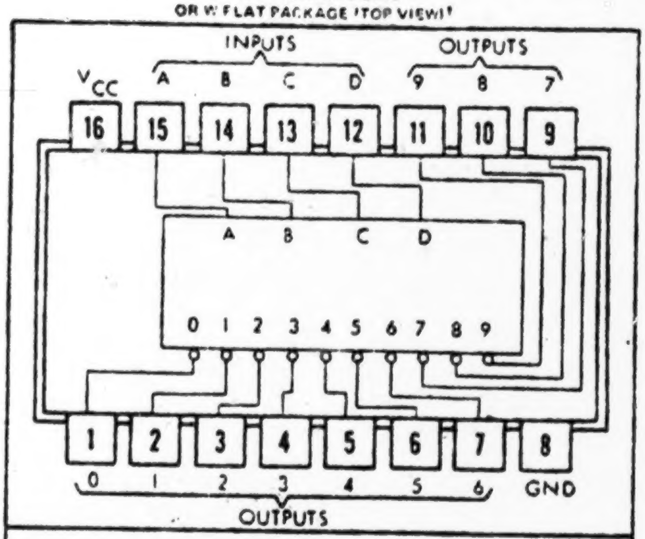
- 4-Line-to-16-Line Decoders
- 3-Line to 8-Line Decoders
- featuring diode-clamped inputs

description

These monolithic decimal decoders consist of eight inverters and ten four-input NAND gates. The inverters are connected in pairs to make BCD input data available for decoding by the NAND gates. Full decoding of valid input logic ensures that all outputs remain off for all invalid input conditions.

The SN5442/SN7442 BCD-to-decimal, SN5443/SN7443 excess-3-to-decimal, and SN5444/SN7444 excess-3-gray-to-decimal decoders feature familiar transistor-transistor-logic (TTL) circuits with inputs and outputs which are compatible for use with other TTL and DTL circuits. D-c noise margins are typically one volt and power dissipation is typically 140 milliwatts. Full fan-out of 10 is available at all outputs.

J OR N DUAL-IN-LINE
OR W FLAT PACKAGE (TOP VIEW)†



positive logic: see truth tables

†Pin assignments for these circuits are the same for all packages.

SN5442/SN7442 BCD INPUT D	C	B	A	SN5443/SN7443 EXCESS 3 INPUT D	C	B	A	SN5444/SN7444 EXCESS 3 GRAY INPUT D	C	B	A	ALL TYPES DECIMAL OUTPUT 0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	1	1	0	0	1	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	0	1	0	0	0	1	1	0	1	0	1	1	1	1	1	1	1	1
0	0	1	0	0	1	0	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	0	0	1	0	1	1	1	1	0	1	1	1	1	1	1
0	1	0	0	0	1	1	1	0	1	0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	1	1	0	0	0	1	1	0	0	1	1	1	1	1	0	1	1	1	1
0	1	1	0	1	0	0	1	1	1	0	1	1	1	1	1	1	1	0	1	1	1
0	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1
1	0	0	0	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	0
1	0	1	0	1	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1
1	1	0	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

TEXAS INSTRUMENTS
INCORPORATED
POST OFFICE BOX 5012 • DALLAS, TEXAS 75222

TTL MSI — CIRCUIT TYPES SN7495A
4-BIT RIGHT-SHIFT LEFT-SHIFT REGISTERS

A TTL MSI PARALLEL-IN PARALLEL-OUT REGISTER

for application as

- N-Bit Serial-To-Parallel Converter
- N-Bit Parallel-To-Serial Converter
- N-Bit Storage Register

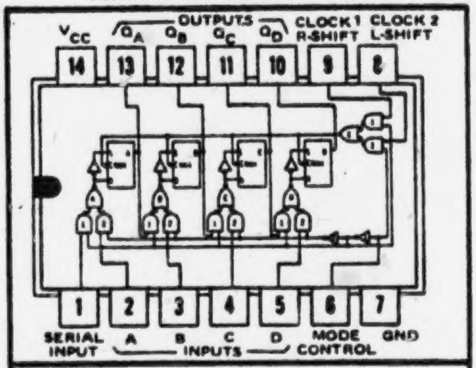
description

This monolithic shift register, utilizing transistor-transistor-logic (TTL) circuits in the familiar Series 54/74 configuration, is composed of four R-S master-slave flip-flops, four AND-OR-INVERT gates, one AND-OR gate, and six inverters-drivers. Internal interconnections of these functions provide a versatile register which will perform right-shift or left-shift operations dependent upon the logical input level to the mode control. A number of these registers may be connected in series to form an n-bit right-shift or left-shift register. This register can also be used as a parallel-in, parallel-out storage register with gate (mode) control.

When a logical 0 level is applied to the mode control input, the number-1 AND gates are enabled and the number-2 AND gates are inhibited. In this mode the output of each flip-flop is coupled to the R-S inputs of the succeeding flip-flop and right-shift operation is performed by clocking at the clock 1 input. In this mode, serial data is entered at the serial input. Clock 2 and parallel inputs A through D are inhibited by the number-2 AND gates.

When a logical 1 level is applied to the mode control input, the number-1 AND gates are inhibited (de-coupling the outputs from the succeeding R-S inputs to prevent right-shift) and the number-2 AND gates are enabled to allow entry of data through parallel inputs A through D and clock 2. This mode permits parallel loading of the register, or with external interconnection, shift-left operation. In this mode, shift-left can be accomplished by connecting the output of each flip-flop to the parallel input of the previous flip-flop (Q_D to input C, and etc.), and serial data is entered at input D.

J OR N
DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic

Mode control = 0 for right shift
Mode control = 1 for left shift or parallel load

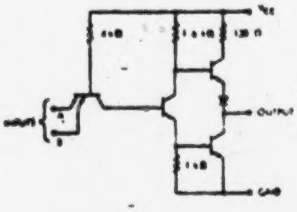
Clocking for the shift register is accomplished through the AND-OR gate E which permits separate clock sources to be used for the shift-right and shift-left modes. If both modes can be clocked from the same source, the clock input may be applied commonly to clock 1 and clock 2. Information must be present at the R-S inputs of the master-slave flip-flops prior to clocking. Transfer of information to the output pins occurs when the clock input goes from a logical 1 to a logical 0.

This shift register is completely compatible with Series 54/74 TTL and DTL logic families. Average power dissipation is typically 195 milliwatts. The SN5495A and SN7495A are unilaterally interchangeable with and replace SN5495 and SN7495, respectively, but offer diode-clamped inputs, improved speed, and reduced power dissipation.

TEXAS INSTRUMENTS
INCORPORATED
POST OFFICE BOX 5012 • DALLAS, TEXAS 75222

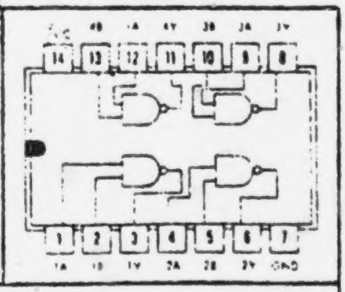
CIRCUIT TYPES SN7400
QUADRUPLE 2-INPUT POSITIVE NAND GATES

schematic (each gate)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: $Y = \overline{AB}$

recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5400 Circuits	4.5	5	5.5	V
	SN7400 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5400 Circuits	−55	25	125	°C
	SN7400 Circuits	0	25	70	°C

electrical characteristics over recommended operating free-air temperature (unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS[†]		MIN	TYP[‡]	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at both input terminals to ensure logical 0 level at output	1			2			V
$V_{in(0)}$	Logical 0 input voltage required at either input terminal to ensure logical 1 level at output	2					0.8	V
$V_{out(1)}$	Logical 1 output voltage	2	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	1	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	3	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	4	V_{CC} = MAX, V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current[§]	5	V_{CC} = MAX	SN5400	−20		−55	mA
				SN7400	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	6	V_{CC} = MAX, V_{in} = 5 V			12	22	mA
$I_{CC(1)}$	Logical 1 level supply current	6	V_{CC} = MAX, V_{in} = 0			4	8	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF,	R_L = 400 Ω		7	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF,	R_L = 400 Ω		11	22	ns

---

[†] For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

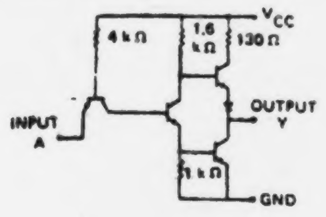
[‡] All typical values are at V_{CC} = 5 V, T_A = 25°C.

[§] Not more than one output should be shorted at a time.

CIRCUIT TYPES SN7404
HEX INVERTERS

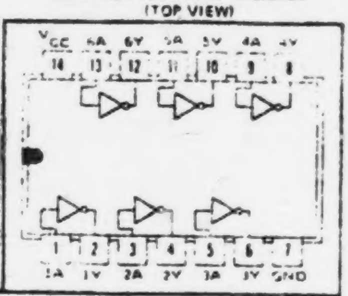
---

schematic (each inverter)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE
(TOP VIEW)



positive logic: $Y = \overline{A}$

recommended operating conditions

	MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}: SN5404 Circuits	4.5	5	5.5	V
SN7404 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N			10	
Operating Free-Air Temperature Range, T_A: SN5404 Circuits	−55	25	125	°C
SN7404 Circuits	0	25	70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS[†]			MIN	TYP[‡]	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at input terminal to ensure logical 0 level at output	15				2			V
$V_{in(0)}$	Logical 0 input voltage required at any input terminal to ensure logical 1 level at output	16						0.8	V
$V_{out(1)}$	Logical 1 output voltage	16	V_{CC} = MIN, I_{load} = −400 μA	V_{in} = 0.8 V,		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	15	V_{CC} = MIN, I_{sink} = 16 mA	V_{in} = 2 V,			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current	18	V_{CC} = MAX,	V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current	18	V_{CC} = MAX,	V_{in} = 2.4 V				40	μA
			V_{CC} = MAX,	V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current[§]	19	V_{CC} = MAX		SN5404	−20		−55	mA
					SN7404	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	20	V_{CC} = MAX,	V_{in} = 5 V			18	33	mA
$I_{CC(1)}$	Logical 1 level supply current	20	V_{CC} = MAX,	V_{in} = 0			6	12	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF,	R_L = 400 Ω		8	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF,	R_L = 400 Ω		12	22	ns

[†] For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

[‡] All typical values are at V_{CC} = 5 V, T_A = 25°C.

[§] Not more than one output should be shorted at a time.

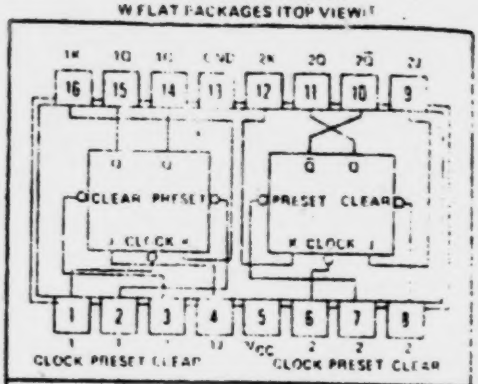
CIRCUIT TYPES SN7476
DUAL J-K MASTER-SLAVE FLIP-FLOPS WITH PRESET AND CLEAR

logic

TRUTH TABLE (Each Flip-Flop)		
t_n		t_{n+1}
J	K	Q
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_n$

NOTES: 1. t_n = Bit time before clock pulse.
2. t_{n+1} = Bit time after clock pulse.

J OR N DUAL-IN-LINE OR W FLAT PACKAGES (TOP VIEW)*


1K
1Q
1Q̄
GND
2K
2Q
2Q̄
2J
16
15
14
13
12
11
10
9
Q̄
Q
CLEAR
PRESET
J CLOCK K
PRESET
CLEAR
K CLOCK J
1
2
3
4
5
6
7
8
1 CLOCK
1 PRESET
1 CLEAR
1J
VCC
2 CLOCK
2 PRESET
2 CLEAR

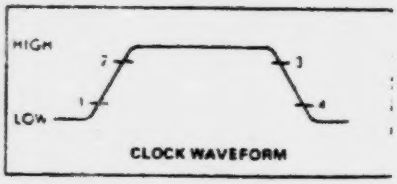
positive logic: Low input to preset sets Q to logical 1.
Low input to clear sets Q to logical 0.
Clear and preset are independent of clock

*Pin assignments for these circuits are the same for all packages.

description

The SN7476 J-K flip-flop is based on the master-slave principle. Inputs to the master section are controlled by the clock pulse. The clock pulse also regulates the state of the coupling transistors which connect the master and slave sections. The sequence of operation is as follows:

1. Isolate slave from master
2. Enter information from J and K inputs to master
3. Disable J and K inputs
4. Transfer information from master to slave.


HIGH
LOW
1
2
3
4

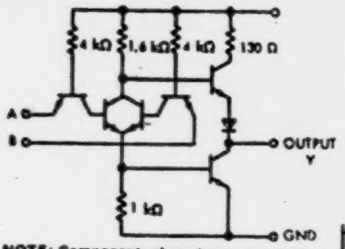
CLOCK WAVEFORM

recommended operating conditions

	MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}: SN5476 Circuits	4.5	5	5.5	V
SN7476 Circuits	4.75	5	5.25	[illegible]
Operating Free-Air Temperature Range, T_A: SN5476 Circuits	-55	25	125	[illegible]
SN7476 Circuits	0	25	70	[illegible]
Normalized Fan-Out From Each Output, N			10	
Width of Clock Pulse, $t_{p(clock)}$ (See figure 69)	20			[illegible]
Width of Preset Pulse, $t_{p(preset)}$ (See figure 70)	25			[illegible]
Width of Clear Pulse, $t_{p(clear)}$ (See figure 70)	25			[illegible]
Input Setup Time, t_{setup} (See figure 69)	$\geq t_{p(clock)}$			
Input Hold Time, t_{hold}	0			

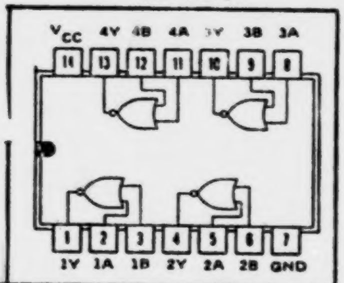
CIRCUIT TYPES SN7402
QUADRUPLE 2-INPUT POSITIVE NOR GATES

schematic (each gate)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: $Y = \overline{A + B}$

recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5402 Circuits	4.5	5	5.5	V
	SN7402 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5402 Circuits	−55	25	125	°C
	SN7402 Circuits	0	25	70	°C

electrical characteristics (over recommended operating free-air temperature range unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at either input terminal to ensure logical 0 level at output	8			2			V
$V_{in(0)}$	Logical 0 input voltage required at both input terminals to ensure logical 1 level at output	9					0.8	V
$V_{out(1)}$	Logical 1 output voltage	9	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	10	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	11	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	12	V_{CC} = MAX, V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current§	13	V_{CC} = MAX	SN5402	−20		−55	mA
				SN7402	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	14	V_{CC} = MAX, V_{in} = 5 V			14	27	mA
$I_{CC(1)}$	Logical 1 level supply current	14	V_{CC} = MAX, V_{in} = 0			8	16	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF, R_L = 400 Ω		8	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF, R_L = 400 Ω		12	22	ns

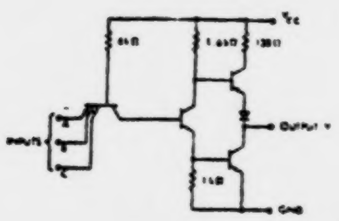
† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.

§ Not more than one output should be shorted at a time.

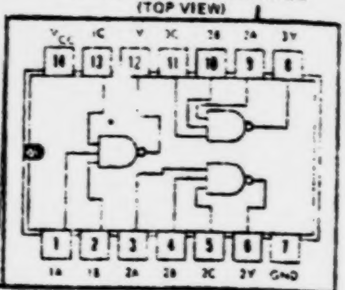
CIRCUIT TYPES SN7410
TRIPLE 3-INPUT POSITIVE NAND GATES

schematic (each gate)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE
(TOP VIEW)



positive logic: $Y = \overline{ABC}$

recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5410 Circuits	4.5	5	5.5	V
	SN7410 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5410 Circuits	−55	25	125	°C
	SN7410 Circuits	0	25	70	°C

electrical characteristics (over recommended operating free-air temperature range unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at all input terminals to ensure logical 0 level at output	1			2			V
$V_{in(0)}$	Logical 0 input voltage required at any input terminal to ensure logical 1 level at output	2					0.8	V
$V_{out(1)}$	Logical 1 output voltage	2	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	1	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	3	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	4	V_{CC} = MAX V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short circuit output current§	5	V_{CC} = 5.. V	SN5410	−20		−55	mA
				SN7410	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	6	V_{CC} = MAX, V_{in} = 5 V			9	16.5	mA
$I_{CC(1)}$	Logical 1 level supply current	6	V_{CC} = MAX, V_{in} = 0			3	6	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF, R_L = 400 Ω		7	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF, R_L = 400 Ω		11	22	ns

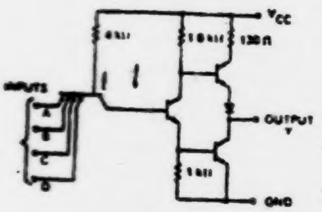
† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.

§ Not more than one output should be shorted at a time.

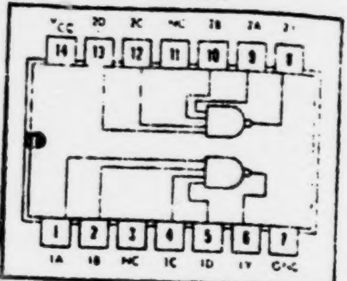
CIRCUIT TYPES SN7420
DUAL 4-INPUT POSITIVE NAND GATES

schematic (each gate)



Component values shown are nominal.
NC—No Internal Connection

J OR N DUAL IN-LINE PACKAGE
(TOP VIEW)



positive logic: $Y = \overline{ABCD}$

recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5420 Circuits	4.5	5	5.5	V
	SN7420 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5420 Circuits	−55	25	125	°C
	SN7420 Circuits	0	25	70	°C

electrical characteristics (over recommended operating free-air temperature range unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS[†]		MIN	TYP[‡]	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at all input terminals to ensure logical 0 level at output	1			2			V
$V_{in(0)}$	Logical 0 input voltage required at any input terminal to ensure logical 1 level at output	2					0.8	V
$V_{out(1)}$	Logical 1 output voltage	2	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	1	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	3	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	4	V_{CC} = MAX, V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current[§]	5	V_{CC} = MAX	SN5420	−20		−55	mA
				SN7420	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	6	V_{CC} = MAX, V_{in} = 5 V			8	11	mA
$I_{CC(1)}$	Logical 1 level supply current	6	V_{CC} = MAX, V_{in} = 0			2	4	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF, R_L = 400 Ω		8	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF, R_L = 400 Ω		12	22	ns

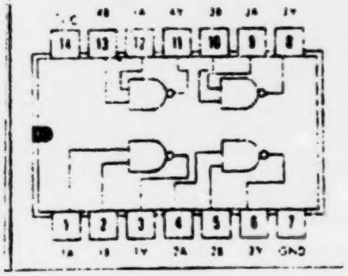
---

[†] For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

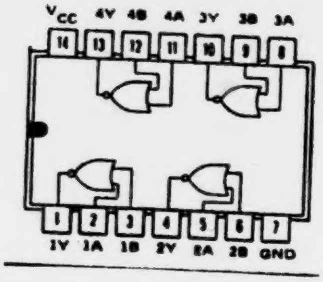
[‡] All typical values are at V_{CC} = 5 V, T_A = 25°C.

[§] Not more than one output should be shorted at a time.

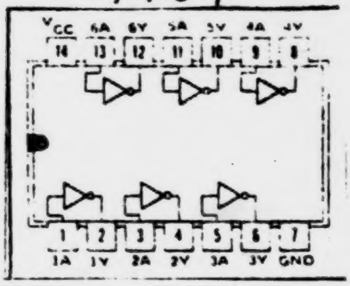
7400


VCC
4B
4A
4Y
3B
3A
3Y
14
13
12
11
10
9
8
1
2
3
4
5
6
7
1A
1B
1Y
2A
2B
2Y
GND

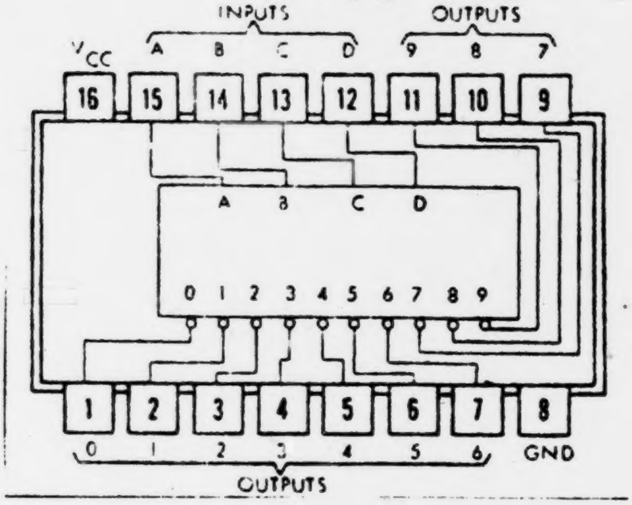
7402


V_{CC}
4Y
4B
4A
3Y
3B
3A
14
13
12
11
10
9
8
1
2
3
4
5
6
7
1Y
1A
1B
2Y
2A
2B
GND

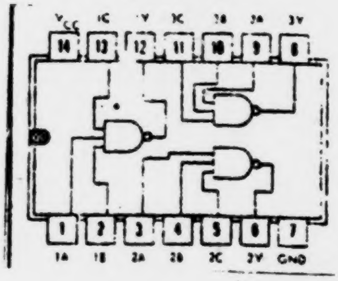
7404


V_{CC}
6A
6Y
5A
5Y
4A
4Y
14
13
12
11
10
9
8
1
2
3
4
5
6
7
1A
1Y
2A
2Y
3A
3Y
GND

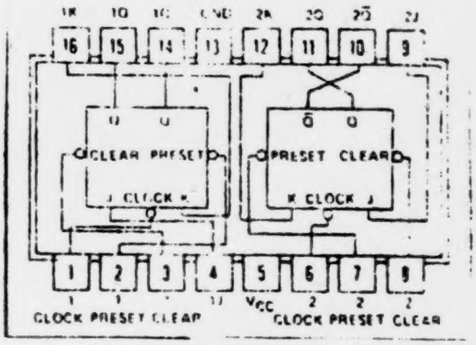
7442


INPUTS
OUTPUTS
V_{CC}
A
B
C
D
9
8
7
16
15
14
13
12
11
10
9
A
B
C
D
0
1
2
3
4
5
6
7
8
9
1
2
3
4
5
6
7
8
0
1
2
3
4
5
6
GND
OUTPUTS

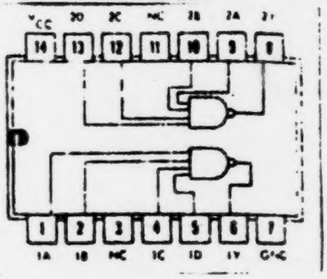
7410


V_{CC}
1C
1Y
3C
3B
3A
3Y
14
13
12
11
10
9
8
1
2
3
4
5
6
7
1A
1B
2A
2B
2C
2Y
GND

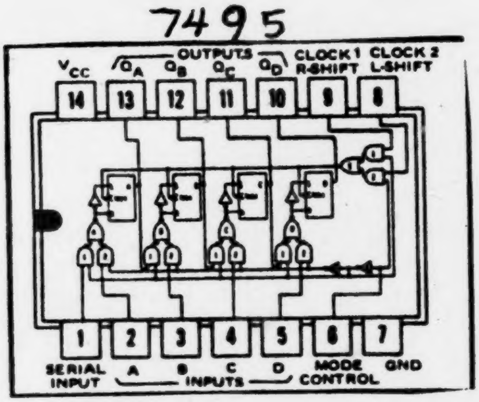
7476


1K
1Q
1Q̄
GND
2K
2Q
2Q̄
2J
16
15
14
13
12
11
10
9
Q̄
Q
CLEAR
PRESET
J
CLOCK
K
Q̄
Q
PRESET
CLEAR
K
CLOCK
J
1
2
3
4
5
6
7
8
1
1
1
1J
V_{CC}
2
2
2
CLOCK
PRESET
CLEAR
CLOCK
PRESET
CLEAR

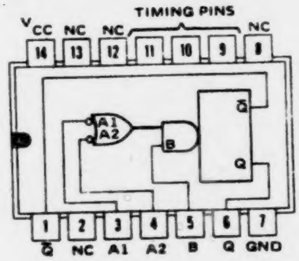
7420


V_{CC}
2D
2C
NC
2B
2A
2Y
14
13
12
11
10
9
8
1
2
3
4
5
6
7
1A
1B
NC
1C
1D
1Y
GND

7495


V_{CC}
OUTPUTS
Q_A
Q_B
Q_C
Q_D
CLOCK 1
R-SHIFT
CLOCK 2
L-SHIFT
14
13
12
11
10
9
8
1
2
3
4
5
6
7
SERIAL
INPUT
A
B
C
D
INPUTS
MODE
CONTROL
GND

74121


TIMING PINS
V_{CC}
NC
NC
NC
14
13
12
11
10
9
8
A1
A2
B
Q̄
Q
1
2
3
4
5
6
7
Q̄
NC
A1
A2
B
Q
GND

LOGIC EXPERIMENT III

USE OF INTEGRATED CIRCUITS IN COMMON COMPUTER HARDWARE DEVICES

Purpose.

The purpose of this experiment is to demonstrate how several different logic components to which you have been introduced either in lecture or in previous experiments can be connected together to make circuits useful in computer applications. We will deal specifically with digital-to-analog and analog-to-digital converters.

You will be provided with a digital-to-analog converter (DAC) chip; by adding components to it, an inexpensive analog-to-digital converter (ADC) can be constructed. After this has been done, modifications can be made, changing the ADC from a counter-converter to a continuous converter. Finally, a different ADC circuit will be built, without using the DAC chip at all.

PART I - Setting up the DAC circuit, and modifying it to make an ADC.

Components Used.

a) Datel DAC-9 - Digital-to-analog converter module.

b) RC741DN - Eight pin operational amplifier used as a current-to-voltage converter.

c) VOM - Volt-ohm-milliammeter used to monitor the DAC and current-to-voltage converters, and to troubleshoot circuits.

d) VRS - Heath voltage reference source, used as analog signal. (CAUTION - never use above 5 Volts)

e) Transistorized Power Supply

Revised, April, 1973

f) LM710CN - Voltage comparator - Puts out a logical 1 or 0, depending on differential voltage at its + and - inputs.

g) SN7400 - Dual input NAND gates - Used to make an R-S flip-flop and to control gating pulses to the counter.

h) SN7404 inverter - Used to convert the R-S flip-flop output for controlling a data storage device (latch).

i) SN7493 binary counter - Used as the source of digital signals for the DAC, and for the representation of analog signals of the ADC.

j) SN7475 - Bistable data latch - Used to store the contents of the counter when the counter's binary contents equal the input analog signal.

k) Resistors - One 2K, ¼ watt resistors for the current-to-voltage converter.

l) Capacitors - Several .012 mfd. decoupling capacitors to be used on all logic chips at + and - 15 Volt inputs to eliminate high frequency noise.

m) Power supply - One ±15 Volt Analog Devices power supply for the 710 and 741 chips (may be built inside the Elite).

Introduction:

Figure 1 shows a schematic of the first circuit you are to wire. We will trace its operation starting with the SN7493 counter.

Initially, the counter should be cleared and the Control FF should be set to a logical "1" output. With the counter at zero, the digital converter puts out zero current. When the Control FF is set, the pulser output can be seen by the counter. As the count increases, so does the DAC output until

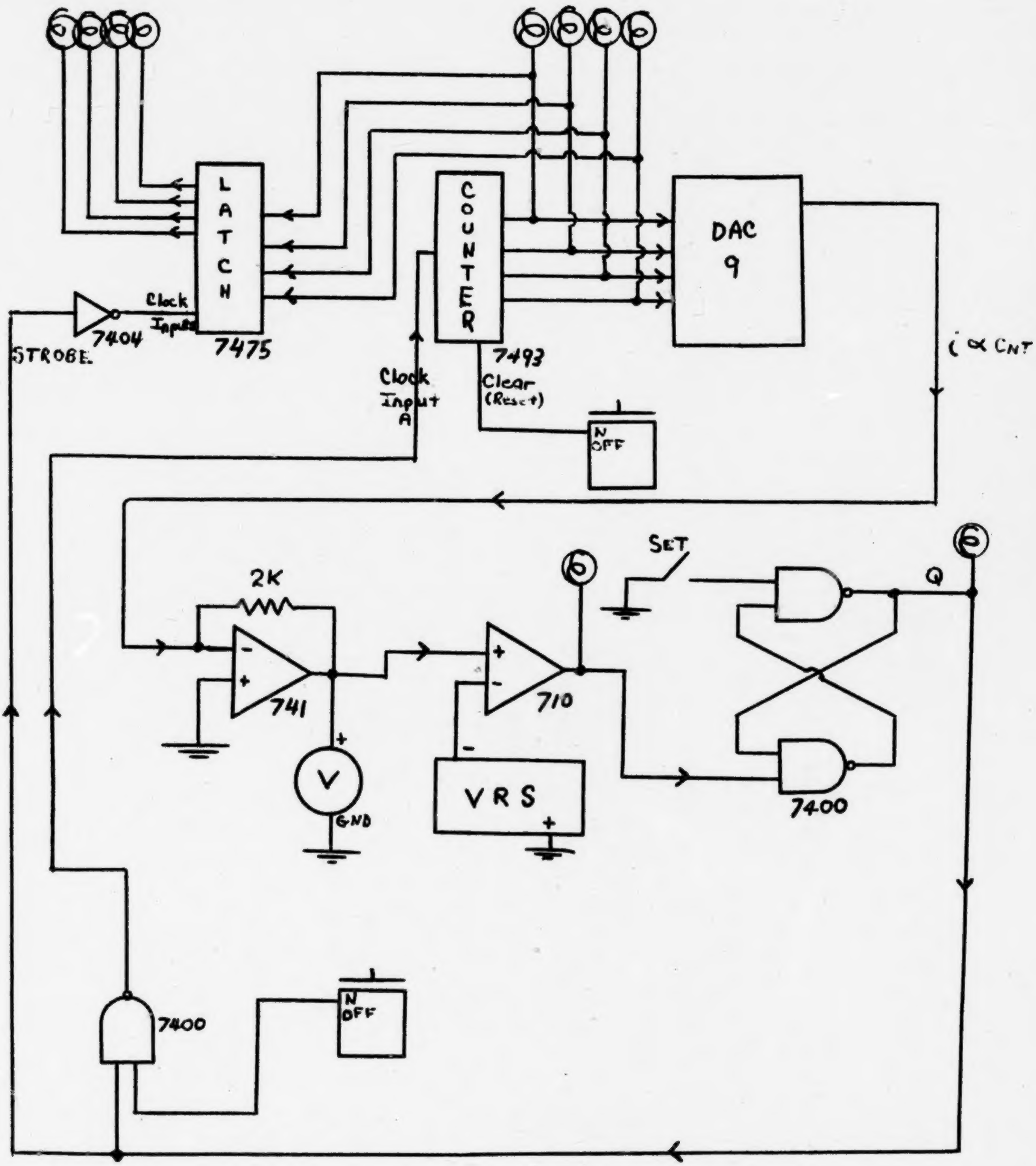

LATCH
7475
Clock Input
7404
STROBE
COUNTER
7493
Clock Input A
Clear (Reset)
N
OFF
DAC 9
i ∝ CNT
2K
741
GND
V
710
VRS
SET
Q
7400
N
OFF
7400

Figure 1. Counter ADC

it reaches a maximum of about 2.5 mA at a count of 15. The DAC output goes through a current-to-voltage converter using a 741 operational amplifier. From there, the signal is led to a 710 voltage comparator. Here, the VRS analog voltage is compared to the converter voltage. The function of the overall circuit is to generate a digital representation of the VRS voltage. This is done by inhibiting the pulses from reaching the counter when $|V_{DAC}| \geq |V_{VRS}|$. The 710 chip as wired will output a logical 1 as long as the $|V_{DAC}| < |V_{VRS}|$ with both inputs having the same polarity. Starting at zero, the V_{DAC} increases until $|V_{DAC}| > |V_{VRS}|$. At this point, the comparator goes to logical 0. This changes the Q output of the R-S flip-flop from 1 to 0. As a result, the pulser output is inhibited at the 7400 NAND gate. Also, a 7475 data latch can be enabled by the inverted flip-flop output. At this point, then, counting is stopped, and the digital representation of the VRS voltage is "strobed" into the latch. It is then stored there, even if counting is continued. By manually resetting the counter and flip-flop, another conversion can be made.

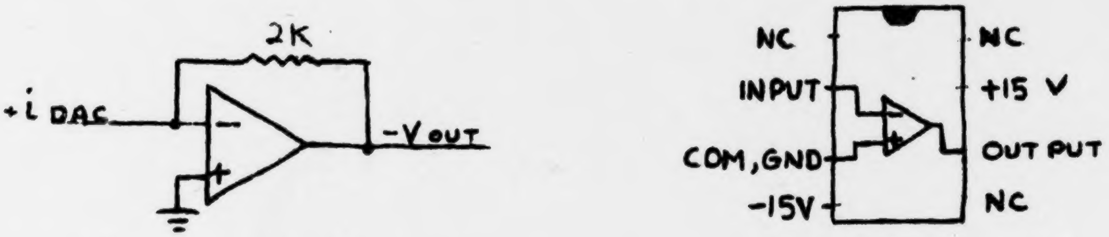
How could you modify the circuit so that another conversion process could be initiated automatically? Could you set up the circuit so that conversions could be made repetitively after a start command?

Experimental Details.

A) Hook up a 7493 counter using the data sheet provided. Hook up the A, B, C, and D outputs to lights. Hook up a norm off pulser to the reset pin 2. Don't forget to connect pin 1 to pin 12. Temporarily connect a norm off pulser to clock input pin 14. Check that the counter counts and resets properly.

B) Power a DAC-9 using pin 9 for +15V D.C. from your power supply and pin 11 for ground. Make sure ground and common are shorted on the power supply. Ground pins 5 through 8. They are for least significant bits and will not be used in this experiment. Hook up the A, B, C, and D counter outputs to pins 4, 3, 2, and 1 of the DAC. Pin 1 is for the most significant bit and pin 4 for the least significant bit. Take the output from pin 12. Connect it to your VOM using the 15 mA scale. With the counter at zero, your converter should give out zero current, and at a count of 15, the output should be about 2.5 mA. Make a plot of mA output vs. count.

C) The current output of the DAC must be converted to a voltage. Wire a 741 op. amp. as shown in Figure 2.



Current-to-voltage Converter

741 Connections

Figure 2

In this mode, $V_{out} \doteq -i_{in} \times R_f$. Since $R_f = 2K$, the maximum output is -5 Volts. Be sure to put decoupling capacitors across +15V to the ground and -15 V to ground for the op. amp. Now plot voltage output from the op. amp. vs. digital counts.

D) The 710 comparator will be damaged if voltages greater than ±7V are present at its inputs or if the difference in voltage between its inputs is greater than 5 Volts. The comparator output should be logical 1, until the $|V_{DAC}| > |V_{VRS}|$. Therefore, the negative VRS signal should be hooked up to the - input of the comparator. Use your data sheet for proper pin assignments. Connect the op. amp. output to the comparator + terminal. Power the 710 chip connecting the +15V power supply to pin 11 and -7V from the transistor power supply to pin 6. Use decoupling capacitors as above.

Clear the counter. Set the VRS to -4V. At this point, the comparator should be at logical 1. Monitor it with an indicator light. Now pulse the counter. At a count of about 12, the DAC will have an approximate output of (-5V x 12/15) = -4 Volts, -5V being the full scale reading corresponding to a count of 15. So the comparator should flip to the 0 state. Verify this, monitoring the V_{DAC} with the VOM. If the light does not go out before a count of 15, recheck your wiring.

E) Because the comparator oscillates near the null point, another component must be included to generate a steady logical 0 at the comparator's transition point. Thus, the cross coupled NAND gate (RS flip-flop) is used in Figure 1 (Control FF). Set the FF by grounding the set input MOMENTARILY, through a toggle switch. Hook the comparator to the reset input. Monitor the Q output of the flip-flop. When the comparator output goes from 1 to 0, this will change the state of the Q output from 1 to 0. Any oscillations at the reset input after the initial transition will not affect the Q output. Check the operation of this gate using a norm off pulser in place of the comparator. Use decoupling capacitors for the 7400's between V_{cc} and GND.

F) Connect the R-S flip-flop's Q output to one input of a 7400 NAND gate. The other input is connected to a norm off clock pulser, which had previously been connected directly to the 7493 counter. The gate output goes to the input of the 7493 counter.

At this point, the ADC converter is completed. To check its operation, reset the counter, and set -3 Volts on the VRS. Make sure the Q output of the flip-flop is at logical 1. Now pulse the counter. It should count up only to that value corresponding to -3V and stop. Continued pulsing is inhibited from reaching the counter through the NAND gate as long as the flip-flop remains at logical 0. Don't worry if the comparitor light stays on. During this time, the contents of the counter can be strobed into a data latch which will store the data even after the counter is reset. Clearing the counter, then setting the flip-flop, will allow repetitive runs.

G) OPTIONAL. Hook uppower and common to a 7475 data latch according to the data sheet. Use decoupling capacitors. Hook up the outputs of the counter pins 12, 9, 8, and 11 to pins 2, 3, 6, and 7 of the latch. Connect pins 16, 15, 10, and 9 of the latch to indicator lights. Take the signal coming from the R-S flip-flop, invert it using a 7404 gate, and connect it to the two clock inputs of the latch, pins 4 and 13. Only when the flip-flop's Q output goes to 0 will data be strobed into the latch. By holding the set input of the flip-flop at ground, the counting can be resumed without the contents of the latch changing from that when the counting was first inhibited.

Plot the latch contents as a function of different VRS voltages from 0 to -5Volts to examine how well your analog-to-digital converter will perform. How good is its resolution?

If you modify the circuit to provide automatic repetitive conversions, be sure to check your proposed circuit with the lab instructor before proceeding to wire it up. What is the conversion time of the ADC for the maximum voltage? minimum voltage?

PART II - Constructing an ADC which continuously follows a changing analog signal.

New Components.

a) 74193 - up/down binary counter.

Introduction.

Since actual analog signals change with time, it is desirable to have an ADC which continously follows the analog signal. This can be accomplished be replacing the 7493 counter with a 74193 up/down counter. See Figure 3.

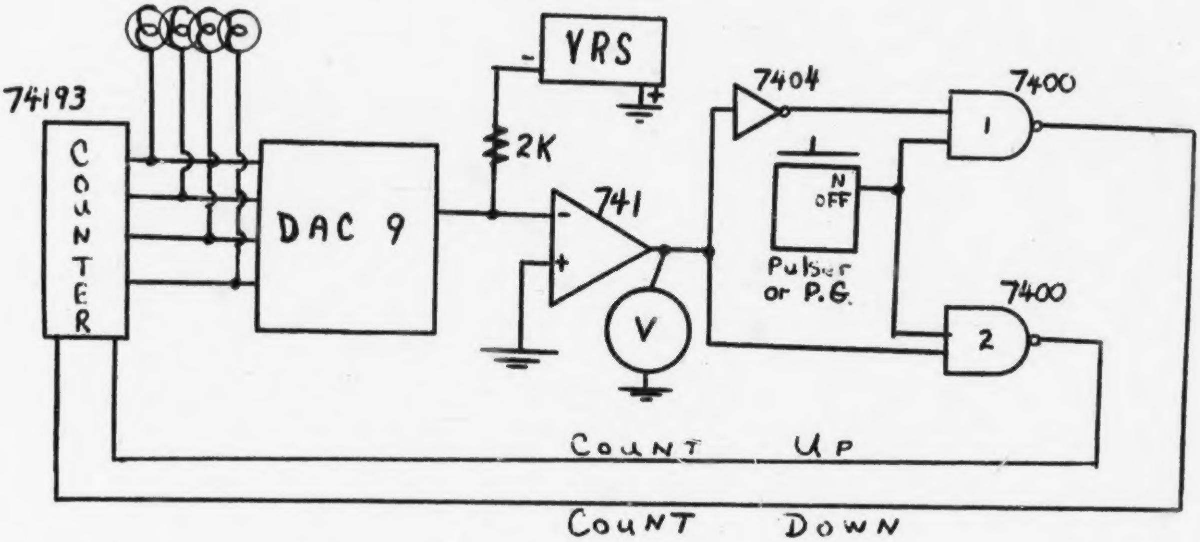

74193
COUNTER
DAC 9
VRS
-
+
2K
741
-
+
V
7404
7400
1
N
OFF
Pulser or P.G.
7400
2
COUNT UP
COUNT DOWN

Figure 3. Continuously Converting ADC

This circuit is somewhat different in operation from the previous one. Now clock pulses are allowed to reach the counter continuously either in the up-count or down-count mode, depending on the output of a 741 op. amp. used as a comparator. In this open loop configuration, the op. amp. theoretically multiplies any voltage differences at its input by about 100,000. However, its output is limited by the ±15V power supply. Note that it is again necessary to use negative analog voltages form the VRS, in order to reach null conditions at the comparator's input. If $|V_{VRS}| > |V_{DAC}|$, then a logical 1 (+15V) will appear at the op. amp's. output since the 741 has greatly amplified and inverted the negative signal at its input. This will allow pulses through the up gate (2), and thus the DAC output will increase until $|V_{VRS}| \approx |V_{DAC}|$ at which point the counter will oscillate ±1 bit. Conversely, if $|V_{VRS}| < |V_{DAC}|$, the op. amp. will put out a logical 0, disabling the up gate while enabling the down gate. Thus, the DAC output will decrease, until $|V_{VRS}| \approx |V_{DAC}|$.

Experimental Details.

A) Hook up the 74193 counter. Pins 7, 6, 2, and 3 are the counter outputs. Monitor them with lights and hook them up to the DAC chip, pins 1 through 4 respectively. Check the operation of the counter by hooking pins 4 and 5 to separate norm on pulsers and pin 14 to a norm off pulser. Pulsing pin 4 causes the chip to count down, pulsing pin 5 causes the chip to count up, while pulsing pin 14 clears the counter. Refer to the supplied data sheet.

B) Hook up the 741 op. amp. in the open loop configuration (no feedback resistor), as shown in Figure 3. Again use the ±15V power supply for the

DAC and op. amp., using decoupling capacitors as needed across ±15 and ground. The VRS red lead is connected to one 2K resistor and this component into pin 2 of the op. amp. The black lead is connected to ground. By using the reverse mode, this unit will then put out the required negative voltages between 0 and -5V. When the VRS is set to -5V, it will generate -2.5 mA, which matches the maximum output current of the DAC. In practice, it may be necessary to set the VRS as high as -7V to balance the maximum DAC output. The DAC output is connected to the (-) input of the 741.

Check the comparator's operation. Set -4V on the VRS and reset the counter. With a norm on pulser connected to pin 5 of the counter and pin 4 left open, the counter output will increase as one pulses it. Initially, the comparator should be logical 1 (+15V). At a count of about 8 or 9, the comparator should go to 0 (-15V). Plot the count at which the comparator changes state for initial VRS settings of -1, -2, -3, -4, and -5 Volts.

C) Now connect the comparator as shown in Figure 3. Use a norm off pulser connected to gates 1 and 2. Connect the down gate output to pin 4 of the counter and the up gate output to pin 5. While continuously pulsing the counter, note how the VRS signal is followed by the digital output as it is varied between 0 and -5V.

One modification which may be made is the substitution of a pulse generator set at about 1 Hz for the hand pulser. Another modification would be to include a data latch to store the counter value at regular intervals. This circuit would then correspond closely to a commercial ADC. Design the modified circuit and check with the lab instructor before implementation.

PART III - Making a ramp type ADC.

New Components.

a) 10 Meg. resistor for ramp circuit.

b) 1 μf capacitor for ramp circuit.

c) 7410 three input NAND gate.

Introduction.

Another type of ADC can be built using a ramp voltage which is compared to the incoming negative analog signal. See Figure 4.

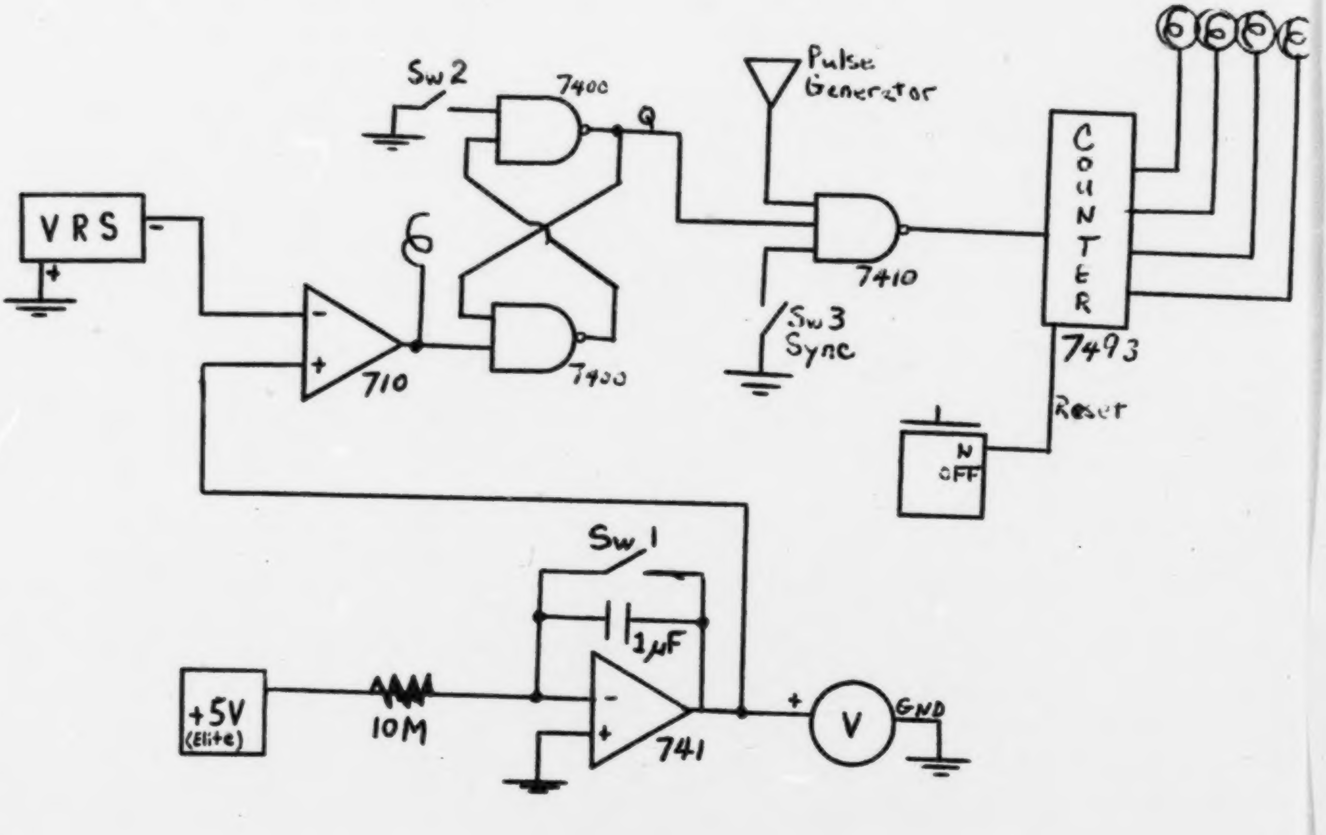

Sw 2
7400
Q
Pulse Generator
VRS
-
+
-
+
710
7400
7410
Sw 3
Sync
COUNTER
7493
Reset
N
OFF
Sw 1
1μF
+5V
(Elite)
10M
-
+
741
+
V
GND

Figure 4. Ramp Type ADC

Here, the 741 operational amp is used to generate a ramp voltage. The ramp will increase until it equals the $|V_{VRS}|$ analog voltage. At that point, the comparator changes form the 1 to 0 state. This resets the R-S flip-flop, putting a 0 on one input of the three-input NAND gate, to which the pulse generator is connected, thus disabling the counter. The larger the VRS signal, the longer it will take for the ramp to equal its value; and the larger will be the counter value when clock pulses are inhibited. The counter value should be proportional to the VRS signal.

Experimental Details.

A) Wire the circuit in Figure 4. The ramp voltage at the output of the 741 integrating amplifier will vary from 0 to -5V over a period of about 10 seconds since

$$e_o = -1/RC \int_o^t e_{in}\, dt$$

Use the +5 Volts on the Elite lab for e_{in} to the integrator. Monitor the ramp output with the VOM. Do not let the ramp voltage go above -7 Volts or the comparator will be damaged. The ramp capacitor can be conveniently shorted using a toggle switch connected across the 1 μf capacitor. Familiarize yourself with this component before hooking it up to the comparator. Use VRS values between 0 and -5 Volts.

B) The remaining components are hooked up as in previous experiments. (Don't forget to use decoupling capacitors.) Monitor the outputs of the comparator and flip-flops with lights.

C) To initiate a run, clear the counter, set the VRS (-) voltage, set the R-S flip-flop by momemtarily closing SW2 , then open the capacitor switch, (SW1), and the sync. switch, (SW3), simultaneously. At the end of a run, close the sync. switch and capacitor switch. The sync. switch while closed will keep clock pulses from reaching the counter. Once open, only the control flip-flop can disable the counter.

The actual readings you obtain for a given VRS signal will depend on the frequency of the pulse generator which must be kept constant for a given series of runs. For example, setting the pulse frequency control at 1 Hz and the vernier at mid scale, a VRS signal of -4 Volts results in a count of about 7.

Plot counter output vs. VRS voltage at two or more clock frequencies.

After doing Logic Experiment IV, you will become familiar with using analog switches. If you have time, you should return to the circuit of Figure 4 and modify the design for automatic repetitive digitization.

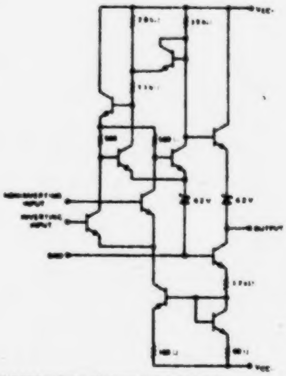
CIRCUIT TYPES SN72710
DIFFERENTIAL COMPARATORS

- Fast Response Times
- Low Offset Characteristics
- Output Compatible with Most TTL and DTL Circuits

description

The SN52710 and SN72710 are monolithic high-speed comparators having differential inputs and a low-impedance output. Component matching, inherent in silicon integrated circuit fabrication techniques, produces a comparator with low-drift and low-offset characteristics. These circuits are especially useful for applications requiring an amplitude discriminator, memory sense amplifier, or a high-speed voltage comparator. The SN52710 is characterised for operation over the full military temperature range of −55°C to 125°C; the SN72710 is characterized for operation from 0°C to 70°C.

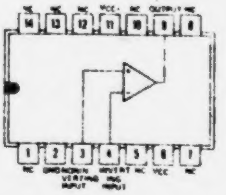
schematic



Component values shown are nominal.

terminal assignments

J OR N
DUAL-IN-LINE PACKAGE (TOP VIEW)



NC—No internal connection

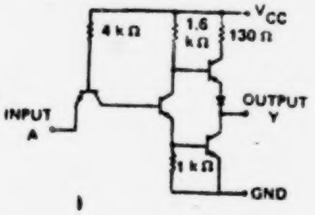
absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

		SN52710	SN72710	UNIT
Supply voltage V_{CC+} (see Note 1)		14	14	V
Supply voltage V_{CC-} (see Note 1)		−7	−7	V
Differential input voltage (see Note 2)		±5	±5	V
Input voltage (either input, see Note 1)		±7	±7	V
Peak output current ($t_w \leq 1$ s)		10	10	mA
Continuous total power dissipation at (or below) 70°C free air temperature (see Note 3)		300	300	mW
Operating free-air temperature range		−55 to 125	0 to 70	°C
Storage temperature range		−65 to 150	−65 to 150	°C
Lead temperature 1/16 inch from case for 60 seconds	J, L, or S package	300	300	°C
Lead temperature 1/16 inch from case for 10 seconds	N package	260	260	°C

NOTES: 1. All voltage values, except differential voltages, are with respect to the network ground terminal.
2. Differential voltages are at the noninverting input terminal with respect to the inverting input terminal.
3. For operation of the SN52710 above 70°C free air temperature, refer to Dissipation Derating Curve, Figure 8.

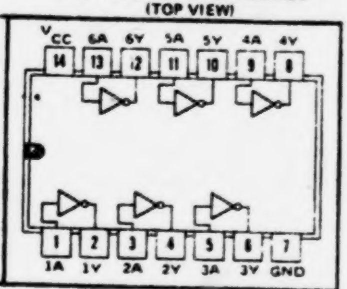
CIRCUIT TYPES SN7404
HEX INVERTERS

schematic (each inverter)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: $Y = \bar{A}$

recommended operating conditions

	MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}: SN5404 Circuits	4.5	5	5.5	V
SN7404 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N			10	
Operating Free-Air Temperature Range, T_A: SN5404 Circuits	−55	25	125	°C
SN7404 Circuits	0	25	70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†			MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at input terminal to ensure logical 0 level at output	15				2			V
$V_{in(0)}$	Logical 0 input voltage required at any input terminal to ensure logical 1 level at output	16						0.8	V
$V_{out(1)}$	Logical 1 output voltage	16	V_{CC} = MIN, I_{load} = −400 μA	V_{in} = 0.8 V,		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	15	V_{CC} = MIN, I_{sink} = 16 mA	V_{in} = 2 V,			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current	18	V_{CC} = MAX,	V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current	18	V_{CC} = MAX,	V_{in} = 2.4 V				40	μA
			V_{CC} = MAX,	V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current§	19	V_{CC} = MAX		SN5404	−20		−55	mA
					SN7404	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	20	V_{CC} = MAX,	V_{in} = 5 V			18	33	mA
$I_{CC(1)}$	Logical 1 level supply current	20	V_{CC} = MAX,	V_{in} = 0			6	12	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF,	R_L = 400 Ω		8	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF,	R_L = 400 Ω		12	22	ns

---

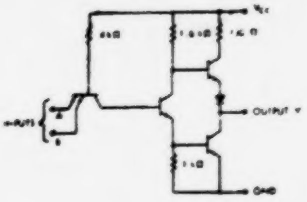
† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.

§ Not more than one output should be shorted at a time.

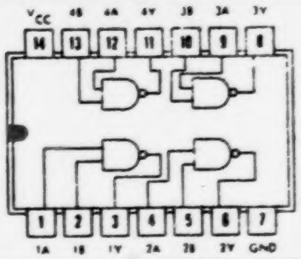
CIRCUIT TYPES SN7400
QUADRUPLE 2-INPUT POSITIVE NAND GATES

schematic (each gate)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE
(TOP VIEW)



positive logic: $Y = \overline{AB}$

recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5400 Circuits	4.5	5	5.5	V
	SN7400 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5400 Circuits	−55	25	125	°C
	SN7400 Circuits	0	25	70	°C

electrical characteristics over recommended operating free-air temperature (unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS[†]		MIN	TYP[‡]	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at both input terminals to ensure logical 0 level at output	1			2			V
$V_{in(0)}$	Logical 0 input voltage required at either input terminal to ensure logical 1 level at output	2					0.8	V
$V_{out(1)}$	Logical 1 output voltage	2	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	1	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	3	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	4	V_{CC} = MAX, V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current[§]	5	V_{CC} = MAX	SN5400	−20		−55	mA
				SN7400	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	6	V_{CC} = MAX, V_{in} = 5 V			12	22	mA
$I_{CC(1)}$	Logical 1 level supply current	6	V_{CC} = MAX, V_{in} = 0			4	8	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF,	R_L = 400 Ω		7	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF,	R_L = 400 Ω		11	22	ns

---

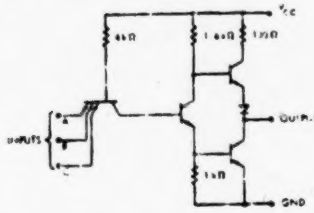
[†] For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

[‡] All typical values are at V_{CC} = 5 V, T_A = 25°C.

[§] Not more than one output should be shorted at a time.

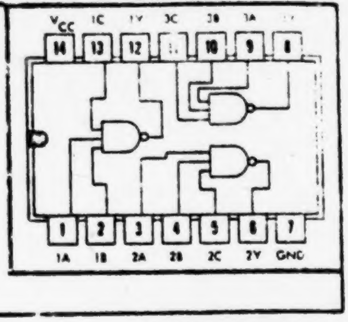
CIRCUIT TYPES SN7410
TRIPLE 3-INPUT POSITIVE NAND GATES

schematic (each gate)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: $Y = \overline{ABC}$

recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5410 Circuits	4.5	5	5.5	V
	SN7410 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5410 Circuits	−55	25	125	°C
	SN7410 Circuits	0	25	70	°C

electrical characteristics (over recommended operating free-air temperature range unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS[†]		MIN	TYP[‡]	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at all input terminals to ensure logical 0 level at output	1			2			V
$V_{in(0)}$	Logical 0 input voltage required at any input terminal to ensure logical 1 level at output	2					0.8	V
$V_{out(1)}$	Logical 1 output voltage	2	V_{CC} = MIN, V_{in} = 0.8 V, I_{load} = −400 μA		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	1	V_{CC} = MIN, V_{in} = 2 V, I_{sink} = 16 mA			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current (each input)	3	V_{CC} = MAX, V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current (each input)	4	V_{CC} = MAX, V_{in} = 2.4 V				40	μA
			V_{CC} = MAX, V_{in} = 5.5 V				1	mA
I_{OS}	Short circuit output current[§]	5	V_{CC} = 5.5 V	SN5410	−20		−55	mA
				SN7410	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	6	V_{CC} = MAX, V_{in} = 5 V			9	16.5	mA
$I_{CC(1)}$	Logical 1 level supply current	6	V_{CC} = MAX, V_{in} = 0			3	6	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF, R_L = 400 Ω		7	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF, R_L = 400 Ω		11	22	ns

---

[†] For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

[‡] All typical values are at V_{CC} = 5 V, T_A = 25°C.

[§] Not more than one output should be shorted at a time.

CIRCUIT TYPES SN7493
4-BIT BINARY COUNTERS

MSI TTL HIGH-SPEED RIPPLE-THROUGH COUNTERS

for applications in

- Digital Computer Systems
- Data-Handling Systems
- Control Systems

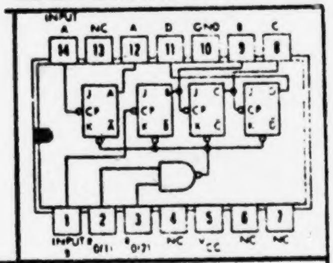
logic

TRUTH TABLE (See Notes 1, 2, and 3)

COUNT	OUTPUT			
	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

NOTES: 1. Output A connected to input B
2. To reset all outputs to logical 0 both $R_{0(1)}$ and $R_{0(2)}$ inputs must be at logical 1.
3. Either (or both) reset inputs $R_{0(1)}$ and $R_{0(2)}$ must be at a logical 0 to count.

J OR N
DUAL-IN-LINE PACKAGE (TOP VIEW)



positive logic: see truth table

NC–No Internal Connection

description

These high-speed, monolithic 4-bit binary counters consist of four master-slave flip-flops which are internally interconnected to provide a divide-by-two counter and a divide-by-eight counter. A gated direct reset line is provided which inhibits the count inputs and simultaneously returns the four flip-flop outputs to a logical 0. As the output from flip-flop A is not internally connected to the succeeding flip-flops the counter may be operated in two independent modes:

1. When used as a 4-bit ripple-through counter, output A must be externally connected to input B. The input count pulses are applied to input A. Simultaneous divisions of 2, 4, 8, and 16 are performed at the A, B, C, and D outputs as shown in the truth table above.
2. When used as a 3-bit ripple-through counter, the input count pulses are applied to input B. Simultaneous frequency divisions of 2, 4, and 8 are available at the B, C, and D outputs. Independent use of flip-flop A is available if the reset function coincides with reset of the 3-bit ripple-through counter.

These circuits are completely compatible with Series 54/74 TTL and DTL logic families. Average power dissipation is 40 mW per flip-flop (160 mW total).

absolute maximum ratings over operating temperature range (unless otherwise noted)

Supply Voltage V_{CC} (See Note 4)		7 V
Input Voltage, V_{in} (See Notes 4 and 5)		5.5 V
Operating Free-Air Temperature Range:	SN5493 Circuits	−55°C to 125°C
	SN7493 Circuits	0°C to 70°C
Storage Temperature Range		−65°C to 150°C

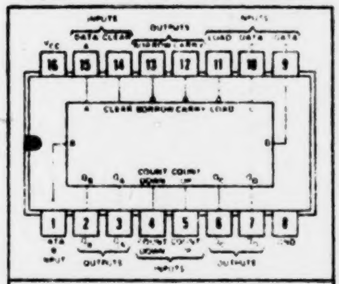
NOTES: 4. These voltage values are with respect to network ground terminal.
5. Input signals must be zero or positive with respect to network ground terminal.

CIRCUIT TYPES ,SN74193
SYNCHRONOUS 4-BIT UP/DOWN COUNTERS (DUAL CLOCK WITH CLEAR

CIRCUIT TYPES SN54192, SN54193, SN74192, SN74193
BULLETIN NO DL-S 7011311, FEBRUARY 1970

- Cascading Circuitry Provided Internally
- Synchronous Operation
- Individual Preset to Each Flip-Flop
- Fully Independent Clear Input
- Typical Maximum Input Count Frequency . . . 32 MHz

J OR N
DUAL-IN-LINE PACKAGE (TOP VIEW)



logic: Low input to load sets $Q_A = A$, $Q_B = B$, $Q_C = C$, and $Q_D = D$

description

These monolithic circuits are synchronous reversible (up/down) counters having a complexity of 55 equivalent gates. The SN54192 and SN74192 are BCD counters and the SN54193 and SN74193 are 4-bit binary counters. Synchronous operation is provided by having all flip-flops clocked simultaneously so that the outputs change coincidently with each other when so instructed by the steering logic. This mode of operation eliminates the output counting spikes which are normally associated with asynchronous (ripple-clock) counters.

The outputs of the four master-slave flip-flops are triggered by a low-to-high-level transition of either count (clock) input. The direction of counting is determined by which count input is pulsed while the other count input is high.

All four counters are fully programmable; that is, the outputs may be preset to any state by entering the desired data at the data inputs while the load input is low. The output will change to agree with the data inputs independently of the count pulses. This feature allows the counters to be used as modulo-N dividers by simply modifying the count length with the preset inputs.

A clear input has been provided which forces all outputs to the low level when a high level is applied. The clear function is independent of the count and load inputs. An input buffer has been placed on the clear, count, and load inputs to lower the drive requirements to one normalized Series 54/74 load. This is important when the output of the driving circuitry is somewhat limited.

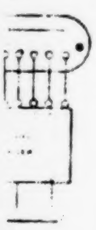
These counters were designed to be cascaded without the need for external circuitry. Both borrow and carry outputs are available to cascade both the up- and down-counting functions. The borrow output produces a pulse equal in width to the count-down input when the counter underflows. Similarly, the carry output produces a pulse equal in width to the count-up input when an overflow condition exists. The counters can then be easily cascaded by feeding the borrow and carry outputs to the count down and count-up inputs respectively of the succeeding counter.

Power dissipation is typically 325 milliwatts for either the decade or binary version. Maximum input count frequency is typically 32 megahertz and is guaranteed to be 25 MHz minimum. All inputs are buffered and represent only one normalized Series 54/74 load. Input clamping diodes are provided to minimize transmission-line effects and thereby simplify system design. The SN54192 and SN54193 are characterized for operation over the full military temperature range of −55°C to 125°C; the SN74192 and SN74193 are characterized for operation from 0°C to 70°C.

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

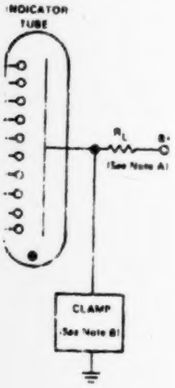
Supply voltage V_{CC} (see Note 1)		7 V
Input voltage (see Note 1)		5.5 V
Operating free-air temperature range:	SN54192 and SN54193 Circuits	−55°C to 125°C
	SN74192 and SN74193 Circuits	0°C to 70°C
Storage temperature range		−65°C to 150°C

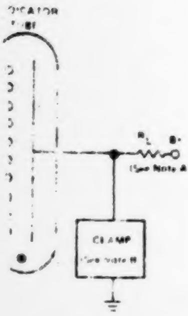
NOTE 1: Voltage values are with respect to network ground terminal.



A

ve decimal 9 may be used
icant bit (MSB) or the
ry 1100) is applied to the
anked, this signal is gated





PRINTED IN U.S.A.
sibility for any circuits shown
free from patent infringement

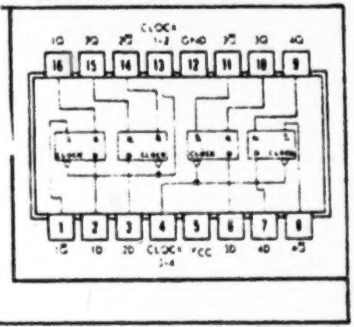
RIGHT TO MAKE CHANGES AT ANY TIME
TO SUPPLY THE BEST PRODUCT POSSIBLE

TTL
MSI

CIRCUIT TYPES ,SN7475,
8-BIT AND 4-BIT BISTABLE LATCHES

SN5477/SN7477
W FLAT PACKAGE

SN5475/SN7475
J OR N DUAL IN-LINE
OR W FLAT PACKAGE (TOP VIEW)†



logic

TRUTH TABLE (Each Latch)	
t_n	t_{n+1}
D	Q
1	1
0	0

NOTES: 1. t_n = bit time before clock negative-going transition.
2. t_{n+1} = bit time after clock negative-going transition.

NC—No internal connection

description

These latches are ideally suited for use as temporary storage for binary information between processing units and input/output or indicator units. Information present at a data (D) input is transferred to the Q output when the clock is high, and the Q output will follow the data input as long as the clock remains high. When the clock goes low, the information (that was present at the data input at the time the transition occurred) is retained at the Q output until the clock is permitted to go high.

The SN5475/SN7475 features complementary Q and $\overline{Q}$ outputs from a 4-bit latch, and is available in the 16-pin packages. For higher component density applications the SN5477/SN7477 4-bit latch is available in the 14-pin flat package. The SN54100/SN74100 features two independent quadruple latches in a single 24-pin dual-in-line package.

These circuits are completely compatible with all popular TTL or DTL families. Typical power dissipation is 40 milliwatts per latch. The Series 54 circuits are characterized for operation over the full military temperature range of −55°C to 125°C and Series 74 circuits are characterized for operation from 0°C to 70°C.

absolute maximum ratings (over operating temperature range unless otherwise noted)

Supply Voltage, V_{CC} (See Note 3)		7 V
Input Voltage, V_{in} (See Notes 3 and 4)		5.5 V
Operating Free-Air Temperature Range:	SN5475 Circuits	−55°C to 125°C
	SN7475 Circuits	0°C to 70°C
Storage Temperature Range		−65°C to 150°C

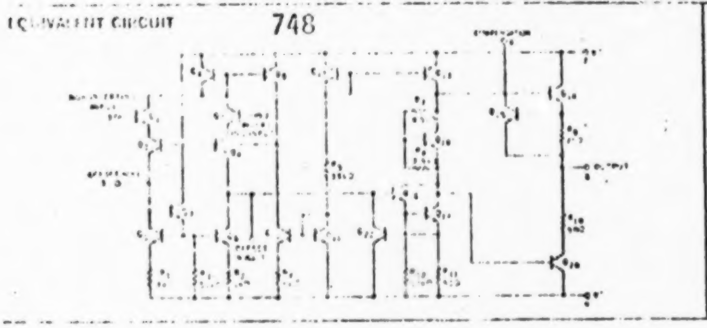
NOTES: 3. These voltage values are with respect to network ground terminal.
4. Input signals must be zero or positive with respect to network ground terminal.

epd ENVIRONMENTAL PRODUCTS

SOLID STATE —— COMPONENTS - KITS - DISPLAYS - APPLICATION NOTES

741 OP AMP

EQUIVALENT CIRCUIT 748


NON-INVERTING INPUT
INVERTING INPUT
COMPENSATION
OUTPUT
OFFSET NULL
V+
V−

CONNECTION DIAGRAM (TOP VIEW)

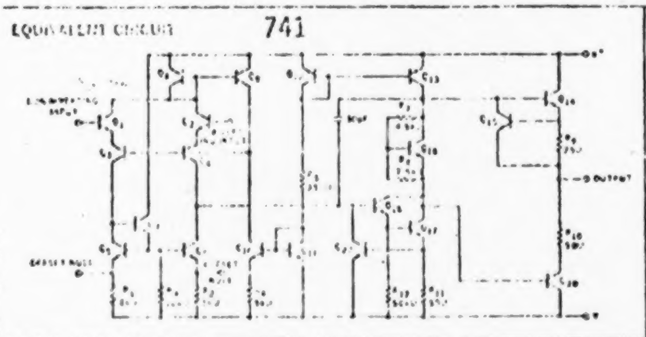
748T

NOTE: Pin 4 connected to case

ABSOLUTE MAXIMUM RATINGS

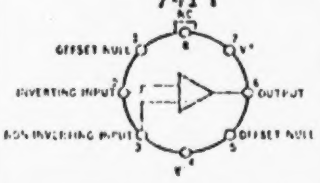
Supply Voltage	± 1
Internal Power Dissipation (Note 1)	500 m
Differential Input Voltage	± 3
Input Voltage (Note 2)	± 1
Storage Temperature Range	−65°C to +15
Operating Temperature Range	0°C to +7
Lead Temperature (Soldering, 60 seconds)	30
Output Short Circuit Duration (Note 3)	Indefi

EQUIVALENT CIRCUIT 741

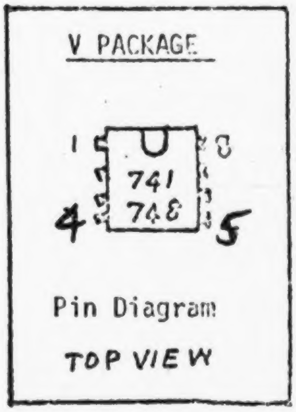

NON-INVERTING INPUT
INVERTING INPUT
OUTPUT
OFFSET NULL
V+
V−

CONNECTION DIAGRAM (TOP VIEW)

741T


NC
8
OFFSET NULL
1
7
V+
INVERTING INPUT
2
6
OUTPUT
NON-INVERTING INPUT
3
OFFSET NULL
5
4
V−

NOTE: PIN 4 CONNECTED TO CASE


V PACKAGE
1
8
741
748
4
5
Pin Diagram
TOP VIEW

ELECTRICAL CHARACTERISTICS ($V_S = \pm 15$ V, $T_A = 25°C$ unless otherwise specified)

PARAMETER	CONDITIONS	MIN.	TYP	MAX.	UNITS
Input Offset Voltage	$R_S \leq 10$ kΩ		2.0	6.0	mV
Input Offset Current			30	200	nA
Input Bias Current			200	500	nA
Input Resistance		0.3	1.0		MΩ
Large Signal Voltage Gain	$R_L \geq 2$ kΩ, $V_{out} = \pm 10$ V	20,000	100,000		
Output Voltage Swing	$R_L \geq 10$ kΩ	±12	±14		V
	$R_L \geq 2$ kΩ	±10	±13		V
Input Voltage Range		±12	±13		V
Common Mode Rejection Ratio	$R_S \leq 10$ kΩ	70	90		dB
Supply Voltage Rejection Ratio	$R_S \leq 10$ kΩ		30	150	μV/V
Power Consumption			50	85	mW
Transient Response (unity gain)	$V_{in} = 20$ mV, $R_L = 2$ kΩ $C_L \leq 100$ pf				
Risetime			0.3		μs
Overshoot			5.0		%
Slew Rate (unity gain)	$R_L > 2$ kΩ		0.5		V/μs
The following specifications apply for $0°C \leq T_A \leq +70°C$:					
Input Offset Voltage	$R_S \leq 10$ kΩ			7.5	mV
Input Offset Current				300	nA
Input Bias Current				800	nA
Large Signal Voltage Gain	$R_L \geq 2$ kΩ, $V_{out} = \pm 10$ V	15,000			
Output Voltage Swing	$R_L \geq 2$ kΩ	±10			V

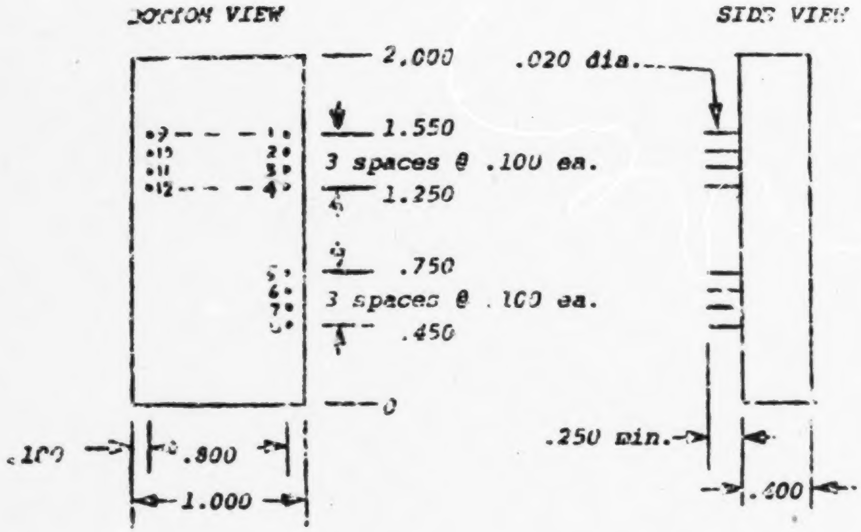
NOTE: The suffix letters T and V refer to the case style. The most common designation for these op-amps is 741C or 748C. In this case the "c" stands for the 0°-70°C temperatu range. Regardless of the case style the specs are identical.

The 748 is identical to t 741 except that no intern compensation capacitor is included.

Addition of the offset co trol is optional. The tw ends of the offset pot ar connected to the offset terminals (1 & 5). The p wiper is connected to p4.

220 South Street - West Lafayette, IN 47906   317-743-4991

DAC - 9


BOTTOM VIEW
SIDE VIEW
2.000
1.550
3 spaces @ .100 ea.
1.250
.750
3 spaces @ .100 ea.
.450
0
.020 dia.
.100
.800
1.000
.250 min.
.400
1
2
3
4
5
6
7
8
9
10
11
12

MODULE INPUT/OUTPUT CONNECTIONS

PIN	FUNCTION	
1	BIT 1 INPUT (MSB)	80
2	BIT 2 INPUT	40
3	BIT 3 INPUT	20
4	BIT 4 INPUT	10
5	BIT 5 INPUT	8
6	BIT 6 INPUT	4
7	BIT 7 INPUT	2
8	BIT 8 INPUT (LSB)	1
9	+15VDC POWER INPUT	
10	REFERENCE TRIM (SEE FIGURE 2, APPLICATIONS PAGE.)	
11	COMMON GROUND (POWER & ANALOG)	
12	ANALOG OUTPUT	

QTY.	ITEM	PART NO.	DESCRIPTION	MFG.	REF. DES.

LIST OF MATERIALS

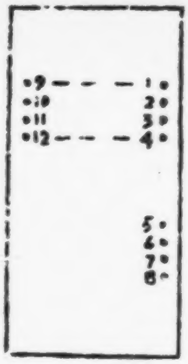
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN INCHES TOLERANCES: DECIMALS .XX ± ___ .XXX ± ___ FRACTIONS ± ___	DRAWN	DATE 11/5/71	VARADYNE SYSTEMS
	CHECKED	DATE Nov 8 71	MECHANICAL LAYOUT & PIN CONNECTIONS OF MODEL DAC-9 SERIES D/A CONVERTER
	PROJ. ENG.	DATE 11-8-71	SIZE A · CODE IDENT. NO. · DWG. NO. 1359 · REV. –
MATERIAL	APPD.	DATE	SCALE · JOB · SHEET 2 OF 3

DAC - 9

MODULE INPUT/OUTPUT CONNECTIONS

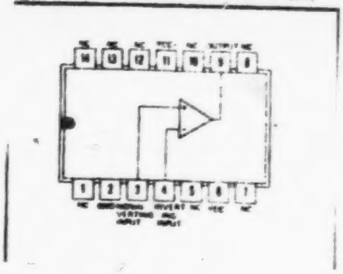
PIN	FUNCTION	
1	BIT 1 INPUT (MSB)	80
2	BIT 2 INPUT	40
3	BIT 3 INPUT	20
4	BIT 4 INPUT	10
5	BIT 5 INPUT	8
6	BIT 6 INPUT	4
7	BIT 7 INPUT	2
8	BIT 8 INPUT (LSB)	1
9	+15VDC POWER INPUT	
10	REFERENCE TRIM (SEE FIGURE 2, APPLICATIONS PAGE.)	
11	COMMON GROUND (POWER & ANALOG)	
12	ANALOG OUTPUT	

BOTTOM VIEW

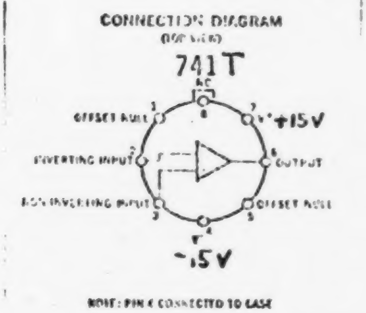

9
10
11
12
1
2
3
4
5
6
7
8

710

J OR N
DUAL-IN-LINE PACKAGE (TOP VIEW)

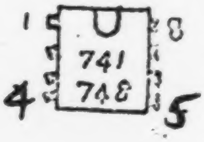

14
13
12
11
10
9
8
1
2
3
4
5
6
7
NC
INVERTING INPUT
NON-INVERTING INPUT
V+
OUTPUT

741


CONNECTION DIAGRAM
(TOP VIEW)
741T
NC
OFFSET NULL
+15V
INVERTING INPUT
OUTPUT
NON-INVERTING INPUT
OFFSET NULL
-15V
1
2
3
4
5
6
7
8

NOTE: PIN 4 CONNECTED TO CASE

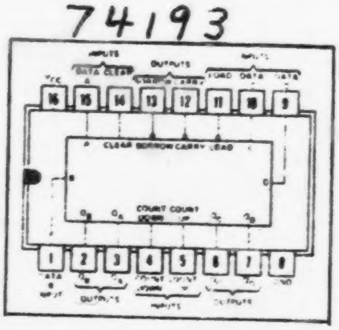
V PACKAGE


1
8
4
5
741
748

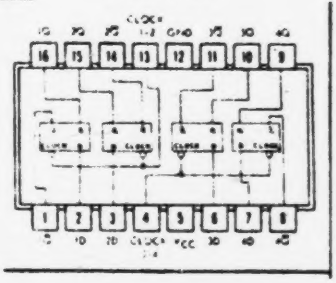
Pin Diagram

TOP VIEW

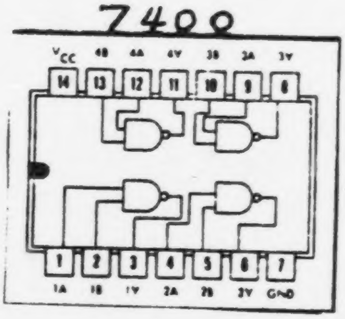
74193


INPUTS
DATA A
CLEAR
OUTPUTS
BORROW
CARRY
INPUTS
LOAD
DATA C
DATA D
VCC
16
15
14
13
12
11
10
9
CLEAR BORROW CARRY LOAD
COUNT DOWN
COUNT UP
1
2
3
4
5
6
7
8
DATA B INPUT
OUTPUTS
INPUTS
OUTPUTS
GND

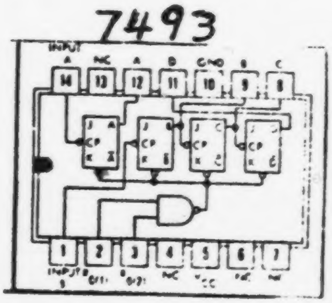
7475


1Q
2Q
2Q
CLOCK 1-2
GND
3Q
3Q
4Q
16
15
14
13
12
11
10
9
1
2
3
4
5
6
7
8
1Q
1D
2D
CLOCK 3-4
VCC
3D
4D
4Q
CLOCK

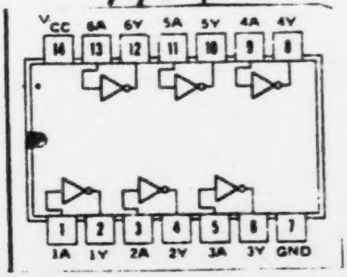
7400


VCC
4B
4A
4Y
3B
3A
3Y
14
13
12
11
10
9
8
1
2
3
4
5
6
7
1A
1B
1Y
2A
2B
2Y
GND

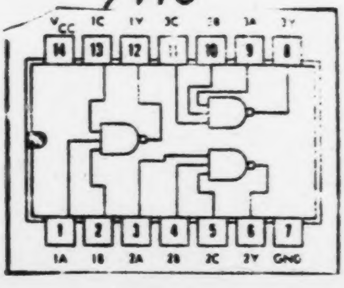
7493


INPUT
A
NC
A
D
GND
B
C
14
13
12
11
10
9
8
J
CP
K
1
2
3
4
5
6
7
INPUT B
R0(1)
R0(2)
NC
VCC
NC
NC

7404


VCC
6A
6Y
5A
5Y
4A
4Y
14
13
12
11
10
9
8
1
2
3
4
5
6
7
1A
1Y
2A
2Y
3A
3Y
GND

7410


VCC
1C
1Y
3C
3B
3A
3Y
14
13
12
11
10
9
8
1
2
3
4
5
6
7
1A
1B
2A
2B
2C
2Y
GND

LOGIC EXPERIMENT IV

INTERFACING ELEMENTS

Materials Needed.

1) Dual Trace Oscilloscope.
2) Elite Logic Lab.
3) ±15V Power Supply. (May be built inside Elite.)
4) Dual Function Generator.
5) 2-SN7490 Decade Counter (see attached data sheet).
6) 1-SN7404 Hex-Inverter (see attached data sheet).
7) 2-CAG 30 FET's and holders (see attached data sheet).
8) 2-741 Op. Amps. (see attached data sheet).
9) 3-100K Resistors.
10) 1-0.1 μf Capacitor.
11) 1-0.01 μf Capacitor.

Introduction.

The purpose of this experiment is to introduce track-and-hold amplifiers and the multiplexer. These are two elements which are commonly used in interfacing analog signals to a digital computer. The multiplexer allows an analog-to-digital converter to be time-shared, thus allowing several different analog inputs to be sampled by one ADC. The track-and-hold amplifier is used to allow precise assignment of the acquisition time of a data point as it should have a very small aperture time. For more details, see Chapters 8, 9 in Reading Material, Volume II.

Revised, April, 1973

Dual Function Generator.

For this experiment, you will be furnished a dual function generator card, which plugs in directly to the Elite lab. In order to be able to use this function generator, it is necessary to supply +15 volts at pin C, -15 volts at pin X, +5 volts at pin A, and ground at pin Z or at the pins indicated on the card. In order to trigger the generator, a positive level (of less than 40 msec.) is applied to pin H. When the generator is triggered, it will output a square wave at pin K, and a triangular wave at pin E. The period of each of these two signals is 100 ms. The signal generator should not be triggered at rates greater than 10 Hz as erratic results will be observed.

Multiplexer.

First, it is necessary to set up the counter circuit composed of two SN7490's as shown in Figure 1 (Note: This part of the circuit will be used throughout the experiment and should not be disassembled.). Apply a pulse train of 1 KHz or slightly less. Monitoring the outputs, you should observe the input frequency, the input frequency divided by 10, and finally the input frequency divided by 100. Now set up the multiplexer circuit as shown in Figure 1. Consult Spec Sheet to be sure of pin connections to CAG30. Now apply the ÷10 output to the multiplexer. Observe the output of the multiplexer and compare it with each of the inputs. Now apply the 1 KHz clock to the multiplexer and again observe the output of the multiplexer. You should note that the higher sampling frequency allows a much better resolution of both signals. In addition, an inversion of the input signals should be noted. Finally, by increasing the scan speed of the oscilloscope, you should be able to observe the speed of the switching from one state to the other and the settling time of the output.

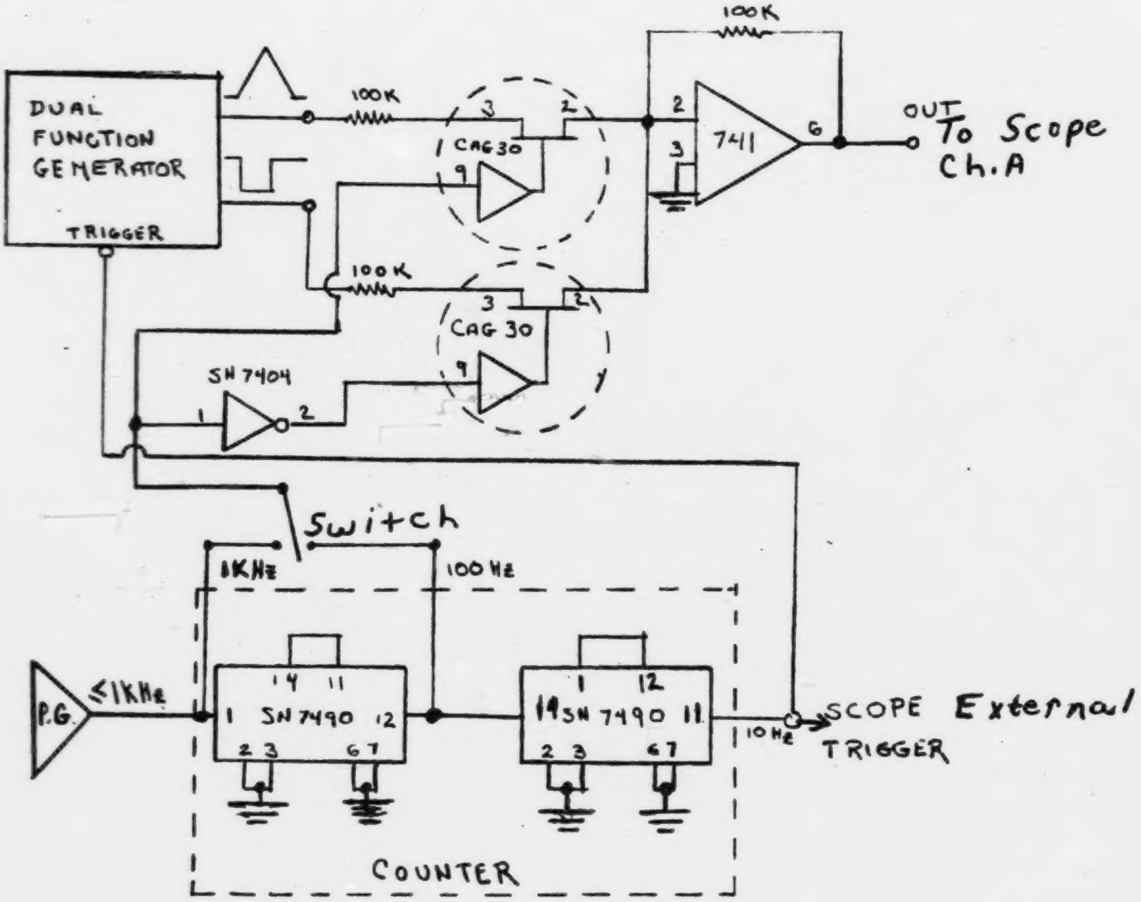
Many commercial multiplexers do not include the amplifier which is used in this circuit. This, however, places much greater restrictions on the FET's which can be used. In order to get a ±10 volt swing, it is necessary to use a FET capable of ±10 volts. In the circuit you have set up, however, the FET is located at the summing point of an amplifier and will not have large signals applied across it. Thus, very inexpensive FET's can be used.

Track-and-Hold Amplifier.

Set up the track-and-hold circuit as shown in Figure 2. Use a 0.1 µf capacitor for the hold capacitor. Apply the 100 Hz clock to the FET. Now observe the input and output of the circuit. You should note a distortion of the input signal. Can you explain this? Now add a second amplifier to the circuit as shown in Figure 3. Again, observe the nature of the input to the system and the output. In this case, no distortion should be noted on the input. Can you explain the difference between this and the earlier circuit?

Now replace the 0.1 µf capacitor with a 0.01 µf capacitor. Observe the output and note the decay of the signal. This is known as the "droop rate" of the track-and-hold.

FIGURE 1. MULTIPLEXER


DUAL FUNCTION GENERATOR
TRIGGER
100K
100K
100K
CAG30
CAG 30
3
2
9
741
2
3
6
OUT
To Scope Ch.A
SN7404
1
2
Switch
1KHz
100 Hz
P.G.
≤1KHz
1
14
11
SN7490
12
2 3
6 7
1
12
14
SN 7490
11
2 3
6 7
10 Hz
SCOPE External TRIGGER
COUNTER

Function Generator
    Pin A = +5V
    Pin C = +15V
    Pin X = -15V
    Pin Z = Ground
741
    Pin 7 = +15V
    Pin 4 = -15V

CAG30
    Pin 1 = +15V
    Pin 7 = -15V
    Pin 8 = +5V
    Pin 10 = Ground
SN7404
    Pin 14 = +5V
    Pin 7 = Ground

SN7490
    Pin 5 = +5V
    Pin 10 = Ground

TRACK & HOLD
CIRCUIT I

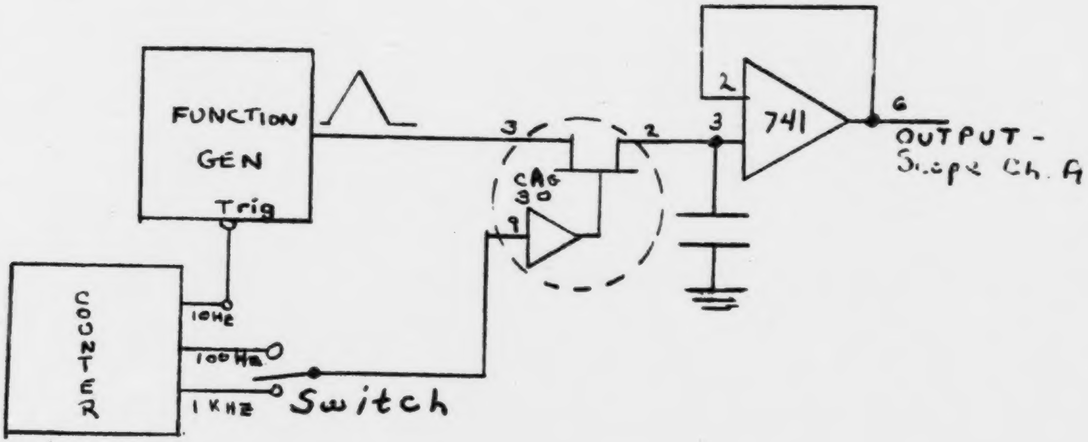

FUNCTION GEN
Trig
COUNTER
10Hz
100Hz
1KHz
Switch
3
2
CAG 30
9
2
3
741
6
OUTPUT - Scope Ch. A

FIGURE 2

CIRCUIT II

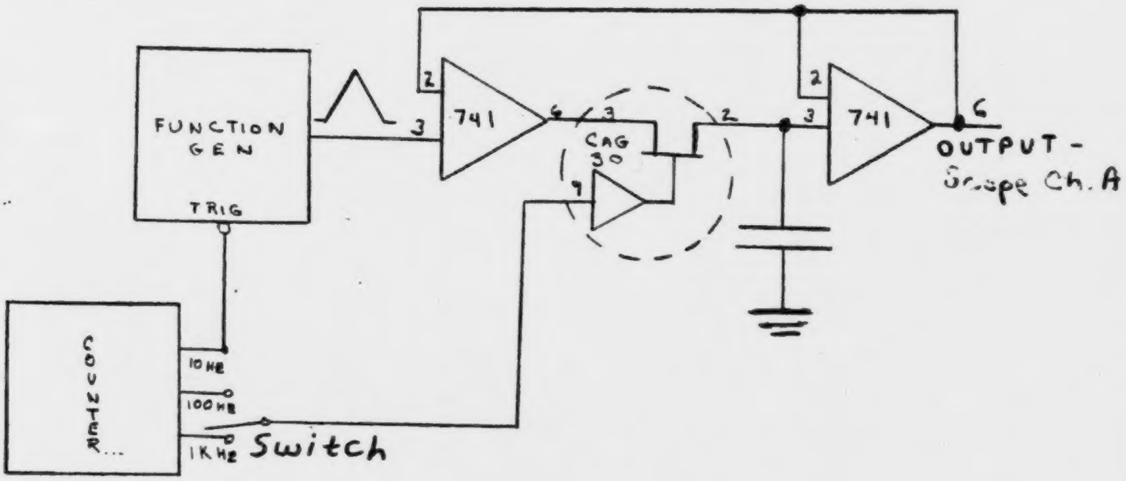
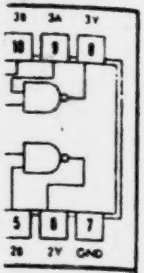

FUNCTION GEN
TRIG
COUNTER...
10Hz
100Hz
1KHz
Switch
2
3
741
6
3
CAG 30
9
2
2
3
741
6
OUTPUT - Scope Ch. A

FIGURE 3

/E PACKAGE
(W)



MAX	UNIT
5.5	V
5.25	V
10	
125	°C
70	°C

wise noted)

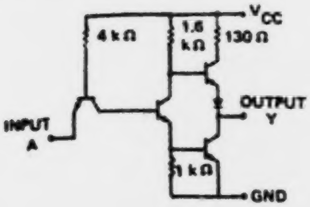
MAX	UNIT
	V
0.8	V
250	µA
0.4	V
−1.6	mA
40	µA
1	mA
22	mA
8	mA

MAX	UNIT
15	ns
45	ns

pplicable device

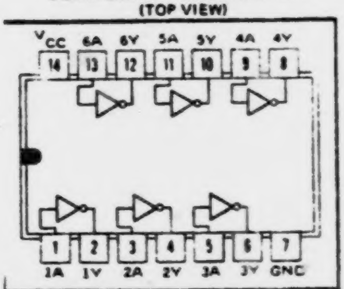
CIRCUIT TYPES SN7404
HEX INVERTERS

schematic (each inverter)



NOTE: Component values shown are nominal.

J OR N DUAL-IN-LINE PACKAGE
(TOP VIEW)



recommended operating conditions

		MIN	NOM	MAX	UNIT
Supply Voltage V_{CC}:	SN5404 Circuits	4.5	5	5.5	V
	SN7404 Circuits	4.75	5	5.25	V
Normalized Fan-Out From Each Output, N				10	
Operating Free-Air Temperature Range, T_A:	SN5404 Circuits	−55	25	125	°C
	SN7404 Circuits	0	25	70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST FIGURE	TEST CONDITIONS†			MIN	TYP‡	MAX	UNIT
$V_{in(1)}$	Logical 1 input voltage required at input terminal to ensure logical 0 level at output	15				2			V
$V_{in(0)}$	Logical 0 input voltage required at any input terminal to ensure logical 1 level at output	16						0.8	V
$V_{out(1)}$	Logical 1 output voltage	16	V_{CC} = MIN, I_{load} = −400 µA	V_{in} = 0.8 V,		2.4	3.3		V
$V_{out(0)}$	Logical 0 output voltage	15	V_{CC} = MIN, I_{sink} = 16 mA	V_{in} = 2 V,			0.22	0.4	V
$I_{in(0)}$	Logical 0 level input current	18	V_{CC} = MAX,	V_{in} = 0.4 V				−1.6	mA
$I_{in(1)}$	Logical 1 level input current	18	V_{CC} = MAX,	V_{in} = 2.4 V				40	µA
			V_{CC} = MAX,	V_{in} = 5.5 V				1	mA
I_{OS}	Short-circuit output current§	1u	V_{CC} = MAX		SN5404	−20		−55	mA
					SN7404	−18		−55	
$I_{CC(0)}$	Logical 0 level supply current	20	V_{CC} = MAX,	V_{in} = 5 V			18	33	mA
$I_{CC(1)}$	Logical 1 level supply current	20	V_{CC} = MAX,	V_{in} = 0			6	12	mA

switching characteristics, V_{CC} = 5 V, T_A = 25°C, N = 10

PARAMETER		TEST FIGURE	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{pd0}	Propagation delay time to logical 0 level	65	C_L = 15 pF,	R_L = 400 Ω		8	15	ns
t_{pd1}	Propagation delay time to logical 1 level	65	C_L = 15 pF,	R_L = 400 Ω		12	22	ns

---

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable device type.

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.

§ Not more than one output should be shorted at a time.

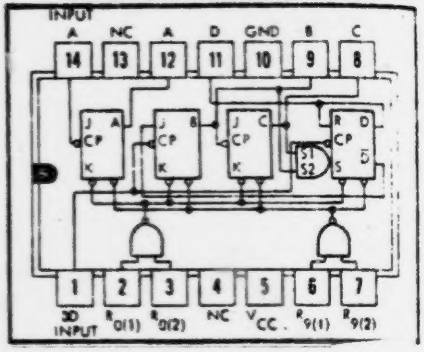
MSI TTL HIGH-SPEED DECADE COUNTERS

for applications in

- **Digital Computer Systems**
- **Data-Handling Systems**
- **Control Systems**

logic

J OR N
DUAL-IN-LINE PACKAGE (TOP VIEW)



TRUTH TABLES

BCD COUNT SEQUENCE
(See Note 1)

COUNT	OUTPUT			
	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

RESET/COUNT (See Note 2)

RESET INPUTS				OUTPUT
$R_{0(1)}$	$R_{0(2)}$	$R_{9(1)}$	$R_{9(2)}$	D C B A
1	1	0	X	0 0 0 0
1	1	X	0	0 0 0 0
X	X	1	1	1 0 0 1
X	0	X	0	COUNT
0	X	0	X	COUNT
0	X	X	0	COUNT
X	0	0	X	COUNT

NC—No Internal Connection

NOTES: 1. Output A connected to input BD for BCD count.
2. X indicates that either a logical 1 or a logical 0 may be present.

description and typical count configurations

These high-speed, monolithic decade counters consist of four dual-rank, master-slave flip-flops internally interconnected to provide a divide-by-two counter and a divide-by-five counter. Gated direct reset lines are provided to inhibit count inputs and return all outputs to zero or to a binary coded decimal (BCD) count of 9. As the output from flip-flop A is not internally connected to the succeeding stages, the count may be separated in three independent count modes:

1. When used as a binary coded decimal decade counter, the BD input must be externally connected to the A output. The A input receives the incoming count, and a count sequence is obtained in accordance with the BCD count sequence truth table shown above. In addition to a conventional zero reset, inputs are provided to reset a BCD count for nine's complement decimal applications.

2. If a symmetrical divide-by-ten count is desired for frequency synthesizers or other applications requiring division of a binary count by a power of ten, the D output must be externally connected to the A input. The input count is then applied at the BD input and a divide-by-ten square wave is obtained at output A.

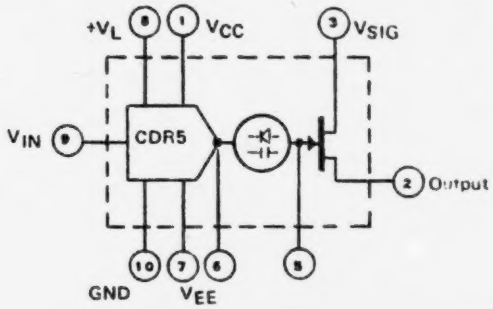
3. For operation as a divide-by-two counter and a divide-by-five counter, no external interconnections are required. Flip-flop A is used as a binary element for the divide-by-two function. The BD input is used to obtain binary divide-by-five operation at the B, C, and D outputs. In this mode, the two counters operate independently; however, all four flip-flops are reset simultaneously.

These circuits are completely compatible with Series 54/74 TTL and DTL logic families. Average power dissipation is 160 mW.

GENERAL PURPOSE FET ANALOG GATE
SPST

CAG30

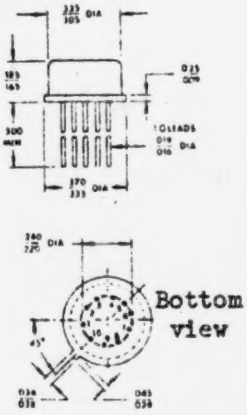
- 60 OHM MAX. RON
- WORKS DIRECTLY FROM LOGIC
- HIGH LOGIC NOISE IMMUNITY
- BREAK BEFORE MAKE ACTION



ABSOLUTE MAXIMUM RATINGS

Operating Temperature	TOPR	–55	to	+125	°C
Storage Temperature	TSTG	–55	to	+150	°C
Logic Supply Voltage	VL	+4.5	to	+5.5	V
Logic Input Voltage	VIN	0	to	VL	V
Positive Supply Voltage	VCC	0	to	+18	V
Negative Supply Voltage	VEE	–18	to	–7	V

TO–100



The CAG30 is a low cost general purpose FET analog gate capable of switching up to ±10V signals directly from DTL or TTL while providing high logic noise immunity (typically 1.0V). The use of a monolithic switching circuit as a driver has the advantage of small size and full military temperature range at low cost. The CAG30 turns off faster than it turns on to allow multiplexing without shorting.

Logic 1 (≥ 2.0V) closes contacts
Logic 0 (≤ 1.0V) opens contacts

ELECTRICAL CHARACTERISTICS: $T_A = +25°C$; $V_L = +5.0V$; $V_B- = -18V$; $V_B+ = +18V$ (Unless otherwise stated)

PARAMETER	SYMBOL	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT
Input Signal Range	V_{SIG}	See Note (1)	–10		+10	V
Drain-Source On-Resistance	Rds	I_D = 1mA, V_{IN} = 2.0V			60	Ohms
Drain-Gate Capacitance	Cdgo	Vdg = 10V, I_S = 0, f = 140KHz		6		pf
Source-Gate Capacitance	Csgo	Vsg = 10V, I_D = 0, f = 140KHz		6		pf
On Current	I_D (on)	V_{DS} = 2.0V, V_{IN} = 2.0V		20		mA
Drain Cut-off Current	I_D (off)	V_{DS} = -10V, V_{IN} = 1.0V			1.0	nA
Source Cut-off	I_S (off)	V_{SD} = -10V, V_{IN} = 1.0V			1.0	nA
Logical "1" Input Voltage	V_{IN} (1)		2.0	1.5		V
Logical "0" Input Voltage	V_{IN} (0)			1.5	1.0	V
Logical "0" Input Current	I_{IN} (0)	V_{IN} = 1.0V			10	μA
Logical "1" Input Current	I_{IN} (1)	V_{IN} = 2.0V			30	μA
Logic Supply Current	I_L			1.8	4	mA
Positive Supply Current	I_{CC}			3.5	5	mA
Negative Supply Current	I_{EE}			3.5	5	mA
Propagation Delay Time to Logical 0	Tpd^0	C_L = 20pf, R_L = 1K			0.5	μS
Propagation Delay Time to Logical 1	Tpd^1	C_L = 20pf, R_L = 1K			1.0	μS
Total Power Dissipation	PT			75	150	mW

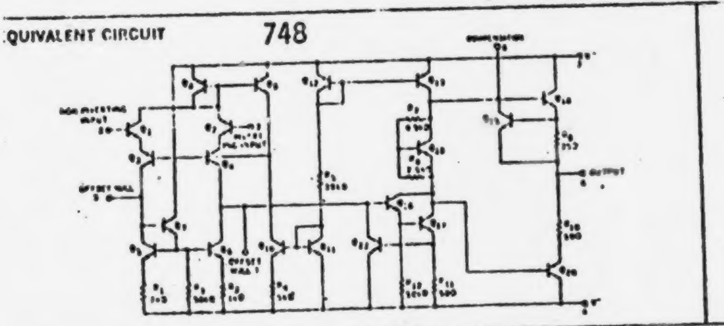
NOTE 1) V_{IN} limits are determined by the V_{CC} and V_{EE} voltages.
V_{IN} min. (V_{EE} + 8V) and V_{IN} max. (V_{CC} -2V)
Ex. V_{EE} = -8V, V_{CC} = +8V, V_{IN} = 0 to +6V

ENVIRONMENTAL PRODUCTS

SOLID STATE —— COMPONENTS · KITS · DISPLAYS · APPLICATION NOTES

741 OP AMP

QUIVALENT CIRCUIT 748



NON-INVERTING INPUT
INVERTING INPUT
OFFSET NULL
OUTPUT
V+
V-

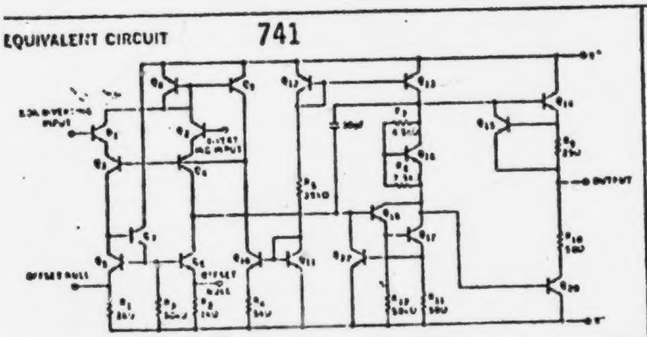
CONNECTION DIAGRAM
(TOP VIEW)
748T

NOTE: Pin 4 connected to case

ABSOLUTE MAXIMUM RATINGS

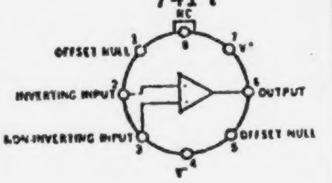
Supply Voltage	±18 V
Internal Power Dissipation (Note 1)	500 mW
Differential Input Voltage	±30 V
Input Voltage (Note 2)	±15 V
Storage Temperature Range	−65°C to +150°C
Operating Temperature Range	0°C to +70°C
Lead Temperature (Soldering, 60 seconds)	300°C
Output Short-Circuit Duration (Note 3)	Indefinite

EQUIVALENT CIRCUIT 741



NON-INVERTING INPUT
INVERTING INPUT
OFFSET NULL
OUTPUT
30pF
V+
V-

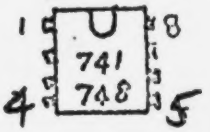
CONNECTION DIAGRAM
(TOP VIEW)
741T



NC
8
OFFSET NULL
1
7
V+
INVERTING INPUT
2
6
OUTPUT
NON-INVERTING INPUT
3
5
OFFSET NULL
4
V-

NOTE: PIN 4 CONNECTED TO CASE

V PACKAGE



1
8
741
748
4
5

Pin Diagram

TOP VIEW

ECTRICAL CHARACTERISTICS ($V_S = \pm15$ V, $T_A = 25$°C unless otherwise specified)

PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNITS
Input Offset Voltage	$R_S \le 10$ kΩ		2.0	6.0	mV
Input Offset Current			30	200	nA
Input Bias Current			200	500	nA
Input Resistance		0.3	1.0		MΩ
Large-Signal Voltage Gain	$R_L \ge 2$ kΩ, $V_{out} = \pm10$ V	20,000	100,000		
Output Voltage Swing	$R_L \ge 10$ kΩ	±12	±14		V
	$R_L \ge 2$ kΩ	±10	±13		V
Input Voltage Range		±12	±13		V
Common Mode Rejection Ratio	$R_S \le 10$ kΩ	70	90		dB
Supply Voltage Rejection Ratio	$R_S \le 10$ kΩ		30	150	μV/V
Power Consumption			50	85	mW
Transient Response (unity gain)	$V_{in} = 20$ mV, $R_L = 2$ kΩ $C_L \le 100$ pF				
Risetime			0.3		μs
Overshoot			5.0		%
Slew Rate (unity gain)	$R_L \ge 2$ kΩ		0.5		V/μs
The following specifications apply for 0°C ≤ T_A ≤ +70°C:					
Input Offset Voltage	$R_S \le 10$ kΩ			7.5	mV
Input Offset Current				300	nA
Input Bias Current				800	nA
Large Signal Voltage Gain	$R_L \ge 2$ kΩ, $V_{out} = \pm10$ V	15,000			
Output Voltage Swing	$R_L \ge 2$ kΩ	±10			V

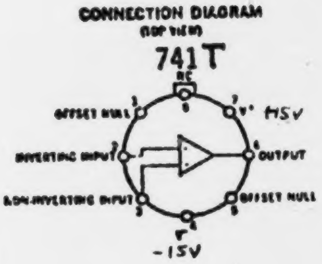
NOTE: The suffix letters T and V refer to the case style. The most common designation for these op-amps is 741C or 748C. In this case the "c" stands for the 0°-70°C temperat range. Regardless of th case style the specs are identical.

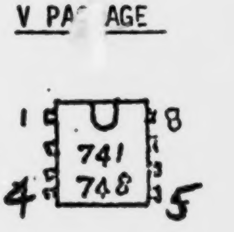
The 748 is identical to 741 except that no inter compensation capacitor i included.

Addition of the offset c trol is optional. The t ends of the offset pot connected to the offset terminals (1 & 5). The wiper is connected to p

220 South Street - West Lafayette, IN 47906 317-743-4991

741

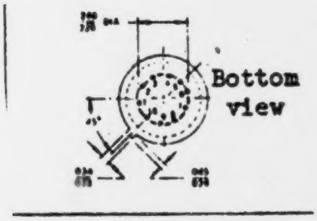

CONNECTION DIAGRAM
(TOP VIEW)
741T
NC
OFFSET NULL
INVERTING INPUT
NON-INVERTING INPUT
V+ +15V
OUTPUT
OFFSET NULL
V- -15V
1
2
3
4
5
6
7
8

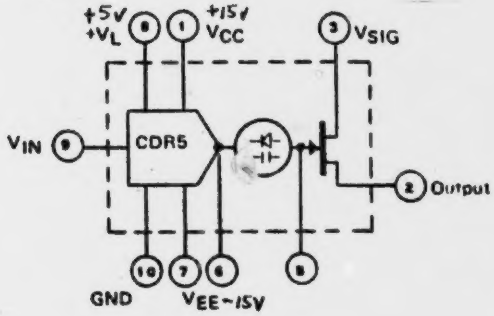

V PACKAGE
1
8
4
5
741
748

Pin Diagram

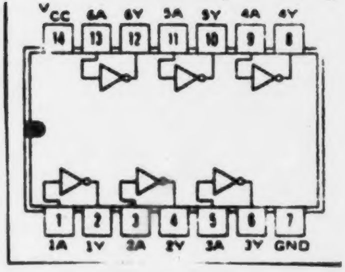
TOP VIEW

CAG-30

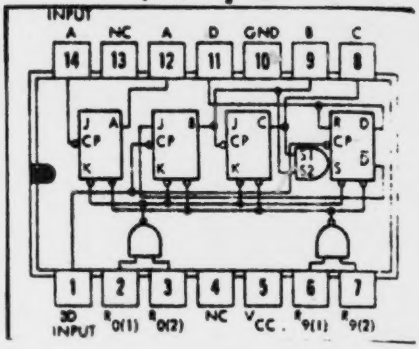

Bottom view


+5V
+VL
+15V
VCC
VSIG
VIN
CDR5
Output
GND
VEE -15V
1
2
3
5
6
7
8
9
10

7404


VCC
6A
6Y
5A
5Y
4A
4Y
14
13
12
11
10
9
8
1
2
3
4
5
6
7
1A
1Y
2A
2Y
3A
3Y
GND

7490


INPUT
A
NC
A
D
GND
B
C
14
13
12
11
10
9
8
J
CP
K
A
B
C
R
D
S1
S2
S
1
2
3
4
5
6
7
BD INPUT
R0(1)
R0(2)
NC
VCC
R9(1)
R9(2)